THEORY AND COMPUTATION OF SPARSE CUTTING PLANES

A Thesis
Presented to
The Academic Faculty

by

Qianyi Wang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Industrial and Systems Engineering

Georgia Institute of Technology
December 2016

THEORY AND COMPUTATION OF SPARSE CUTTING PLANES

Approved by:

Professor Santanu Dey, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor George Nemhauser
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Shabbir Ahmed
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Andy Sun
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Marco Molinaro
Pontifical Catholic University of Rio de Janeiro
Georgia Institute of Technology

Date Approved: August 2016

To my parents.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisors Prof. Santanu Dey and Prof. Marco Molinaro for the continuous support and guidance of my Ph.D. study, for their patience, professional attitude and immense knowledge. I not only learned how to be an excellent researcher, but also learned how to be a better person. I could not have imagined having a better advisor and mentor for my Ph.D. study.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Nemhauser, Prof. Ahmed, and Prof. Sun. They spared their precious time attending my defense and made insightful comments on my thesis as well as my research.

Last but not the least, I would like to thank my friends and family. I will always miss these years I spent with my friends in Atlanta, which is a precious memory of mine. And I am really appreciating the support from my parents. Without their encouragement, I could never achieve this goal.

Contents

List of Tables

List of Figures

SUMMARY

A cutting plane is a linear inequality that is valid for all the feasible solutions of an integer program but may separate some points of the linear programming relaxation. Most state-of-the-art integer programming solvers tend to bias their cutting plane selection towards sparse cuts. In this thesis, we conduct a comprehensive study of sparse cutting planes. We also develop a new approximation algorithm for sparse integer programs.

In the first chapter, we discuss the quality of approximating integer hull by adding sparse cutting planes. Given a polytope P (e.g. the integer hull of a MIP), let P^k be its best approximation using cuts with at most k non-zero coefficients. We consider $\mathrm{d}(P, P^k) = \max_{x \in P^k} (\min_{y \in P} \|x - y\|)$ as a measure of the quality of sparse cuts. In our first result, we present general upper bounds on $\mathrm{d}(P, P^k)$ which depend on the number of vertices in the polytope and exhibit three phases as k increases. Our bounds imply that if P has polynomially many vertices, using half sparsity already approximates it very well. Second, we present a lower bound on $\mathrm{d}(P, P^k)$ for random polytopes that show that the upper bounds are quite tight. Third, we show that for a class of hard packing IPs, sparse cutting-planes do not approximate the integer hull well. Finally, we show that using sparse cutting-planes in extended formulations is at least as good as using them in the original polyhedron, and give an example where the former is actually much better.

In the second chapter, we present an analysis of the strength of sparse cutting-planes for mixed integer linear programs (MILP) with sparse formulations. We examine three kinds of problems: packing problems, covering problems, and more general MILPs with the only assumption that the objective function is non-negative. Given

a MILP instance of one of these three types, assume that we decide on the support of cutting-planes to be used and the strongest inequalities on these supports are added to the linear programming relaxation. Call the optimal objective function value of the linear programming relaxation together with these cuts as z^{cut}. We present bounds on the ratio of z^{cut} and the optimal objective function value of the MILP that depends only on the sparsity structure of the constraint matrix and the support of sparse cuts selected, that is, these bounds are completely data independent. These results also shed light on the strength of scenario-specific cuts for two stage stochastic MILPs.

In the last chapter, we present a new approximation algorithm for solving packing integer programs whose constraint matrix exhibit global sparsity pattern that is known in advance. The algorithm runs in two phases. In the first phase, the sparse packing problem is partitioned into smaller parts and then these small integer programs are solved. In the second phase, the optimal solutions of the smaller problems are patched together into a feasible solution for the original problem by exploiting the sparsity structure of the constraint matrix. We present a theoretical guarantee on the quality of the solution produced by this algorithm, which we show depends only on the sizes of the smaller IPs and the sparsity structure, being otherwise completely data independent. Finally, we experiment on randomly generated large-scale sparse set packing instances and also set-packing integer programs with tree structure and compare the results of our algorithm to those by `CPLEX` and also specialized heuristic for packing integer programs based on the state-of-the-art Greedy Randomized Adaptive Search Procedure (GRASP) of [15]. Our results indicate that on very sparse instances our algorithm shows significant promise over other methods.

Chapter I

APPROXIMATING POLYHEDRA WITH SPARSE INEQUALITIES

1.1 Introduction

Most successful mixed integer linear programming (MILP) solvers are based on branch-&-bound and cutting-plane (cut) algorithms. Since MILPs belong to the class of NP-hard problems, one does not expect the size of branch-&-bound tree to be small (polynomial is size) for every instance. In the case where the branch-&-bound tree is not small, a large number of linear programs must be solved. It is well known that dense cutting-planes are difficult for linear programming solvers to handle. Therefore, most commercial MILPs solvers consider sparsity of cuts as an important criterion for cutting-plane selection and use [21, 2, 34].

Surprisingly, very few studies have been conducted on the topic of sparse cutting-planes. Apart from cutting-plane techniques that are based on generation of cuts from single rows (which implicitly lead to sparse cuts if the underlying row is sparse), to the best of our knowledge only the paper [4] explicitly discusses methods to generate sparse cutting-planes.

The use of sparse cutting-planes may be viewed as a compromise between two competing objectives. As discussed above, on the one hand, the use of sparse cutting-planes aids in solving the linear programs encountered in the branch-&-bound tree faster. On the other hand, it is possible that 'important' facet-defining or valid inequalities for the convex hull of the feasible solutions are dense and thus without adding these cuts, one may not be able to attain significant integrality gap closure. This may lead to a larger branch-&-bound tree and thus result in the solution time to increase.

It is challenging to simultaneously study both the competing objectives in relation to cutting-plane sparsity. Therefore, a first approach to understanding usage of sparse cutting-planes is the following: *If we are able to separate and use valid inequalities with a given level of sparsity (as against completely dense cuts), how much does this cost in terms of loss in closure of integrality gap?*

Considered more abstractly, the problem reduces to a purely geometric question: Given a polytope P (which represents the convex hull of feasible solutions of a MILP), how well is P approximated by the use of sparse valid inequalities. In this chapter we will study polytopes contained in the $[0,\ 1]^n$ hypercube. This is without loss of generality since one can always translate and scale a polytope to be contained in the $[0,\ 1]^n$ hypercube.

1.1.1 Preliminaries

A cut $ax \leq b$ is called *k-sparse* if the vector a has at most k nonzero components. Given a set $P \subseteq \mathbb{R}^n$, define P^k as the best outer-approximation obtained from k-sparse cuts, that is, it is the intersection of all k-sparse cuts valid for P.

For integers k and n, let $[n] := \{1, \ldots, n\}$ and let $\binom{[n]}{k}$ be the set of all subsets of $[n]$ of cardinality k. Given a k-subset of indices $I \subseteq [n]$, define $\mathbb{R}^{\bar{I}} = \{x \in \mathbb{R}^n : x_i = 0 \text{ for all } i \in I\}$. An equivalent and handy definition of P^k is the following: $P^k = \bigcap_{I \in \binom{[n]}{k}} \left(P + \mathbb{R}^{\bar{I}}\right)$. Thus, if P is a polytope, P^k is also a polytope.

1.1.2 Measure of Approximation

There are several natural measures to compare the quality of approximation provided by P^k in relation to P. For example, one may consider objective value ratio: maximum over all costs c of expression $\frac{z^{c,k}}{z^c}$, where $z^{c,k}$ is the value of maximizing c over P^k, and z^c is the same for P. We discard this ratio, since this ratio can become infinity and not provide any useful information[1]. Similarly, we may compare the volumes of P and P^k. However, this ratio is not useful if P is not full-dimensional and P^k is.

In order to have a useful measure that is well-defined for all polytopes contained in $[0,1]^n$, we consider the following *distance measure*:

$$\mathrm{d}(P, P^k) := \max_{x \in P^k} \left(\min_{y \in P} \|x - y\| \right),$$

where $\|\cdot\|$ is the ℓ_2 norm. It is easily verified that there is a vertex of P^k attaining the maximum above. Thus, alternatively the distance measure can be interpreted as the Euclidean distance between P and the farthest vertex of P^k from P.

Observation 1 ($\mathrm{d}(P, P^k)$ is an upper bound on depth of cut)**.** *Suppose $\alpha x \leq \beta$ is a valid inequality for P where $\|\alpha\| = 1$. Let the* depth *of this cut be the smallest $\gamma \geq 0$ such that $\alpha x \leq \beta + \gamma$ is valid for P^k. It is straightforward to verify that $\gamma \leq d(P, P^k)$. Therefore, the distance measure gives an upper bound on additive error when optimizing a (normalized) linear function over P and P^k.*

Observation 2 (Comparing $\mathrm{d}(P, P^k)$ to $\sqrt{n}$)**.** *Notice that the largest distance between any two points in the $[0,\ 1]^n$ hypercube is at most $\sqrt{n}$. Therefore in the rest of the chapter we will compare the value of $d(P, P^k)$ to $\sqrt{n}$.*

1.1.3 Some Examples

In order to build some intuition we begin with some examples in this section. Let $P := \{x \in [0,\ 1]^n : ax \leq b\}$ where a is a non-negative vector. It is straightforward to verify that in this case, $P^k := \{x \in [0,\ 1]^n : a^I x \leq b\ \forall I \in \binom{[n]}{k}\}$, where $a^I_j := a_j$ if $j \in I$ and $a^I_j = 0$ otherwise.

Example 1: Consider the simplex $P = \{x \in [0,1]^n : \sum_{i=1}^n x_i \leq 1\}$. Using the above observation, we have that $P^k = \mathrm{conv}\{e^1, e^2, \ldots, e^n, \frac{1}{k}e\}$, where e^j is the unit vector in

[1] Take $P = \mathrm{conv}\{(0,0), (0,1), (1,1)\}$ and compare with P^1 wrt $c = (1, -1)$.

the direction of the j^{th} coordinate and e is the all ones vector. Therefore the distance measure between P and P^k is $\sqrt{n}(\frac{1}{k} - \frac{1}{n}) \approx \frac{\sqrt{n}}{k}$, attained by the points $\frac{1}{n}e \in P$ and $\frac{1}{k}e \in P^k$. This is quite nice because with $k \approx \sqrt{n}$ (which is pretty reasonably sparse) we get a constant distance. Observe also that the *rate of change of the distance measure* follows a 'single pattern' - we call this a *single phase example*. See Figure 2(a) for $\mathrm{d}(P, P^k)$ plotted against k (in blue) and $k \cdot \mathrm{d}(P, P^k)$ plotted against k (in green).

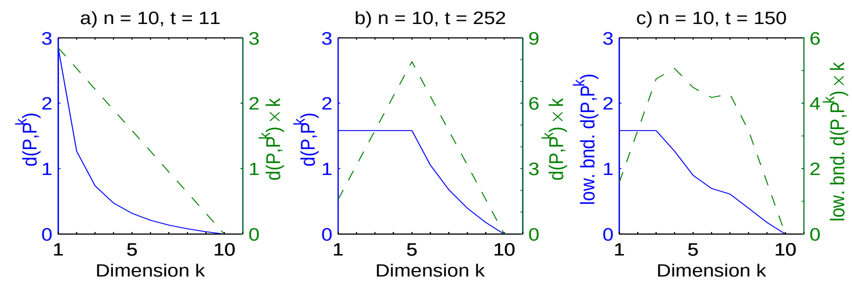


Figure 1: Some examples: (a) Sparsity is good. (b) Sparsity is not so good. (c) Example with three phases.

Example 2: Consider the set $P = \{x \in [0,1]^n : \sum_i x_i \leq \frac{n}{2}\}$. We have that $P^k := \{x \in [0,\ 1]^n : \sum_{i\in I} x_i \leq \frac{n}{2},\ \forall I \in \binom{[n]}{k}\}$. Therefore, for all $k \in \{1, \ldots, n/2\}$ we have $P^k = [0,\ 1]^n$ and hence $\mathrm{d}(P, P^k) = \sqrt{n}/2$. Thus, we stay with distance $\Omega(\sqrt{n})$ (the worst possible for polytopes in $[0,1]^n$) even with $\Theta(n)$ sparsity. Also observe that for $k > \frac{n}{2}$, we have $\mathrm{d}(P, P^k) = \frac{n\sqrt{n}}{2k} - \frac{\sqrt{n}}{2}$. Clearly the rate of change of the distance measure has *two phases*, first phase of k between 1 and $\frac{n}{2}$ and the second phase of k between $\frac{n}{2}$ and n. See Figure 2(b) for the plot of $\mathrm{d}(P, P^k)$ against k (in blue) and of $k \cdot \mathrm{d}(P, P^k)$ against k (in green).

Example 3: We present an experimental example in dimension $n = 10$. The polytope P is now set as the convex hull of 150 binary points randomly selected from the hyperplane $\{x \in \mathbb{R}^{10} : \sum_{i=1}^{10} x_i = 5\}$. We experimentally computed lower bounds on $\mathrm{d}(P, P^k)$ which are plotted in Figure 2(c) as the blue line (for details on this computation see Section A.2 of the appendix). Notice that there are now three phases, which are more discernible in the plot between the lower bound on $k \cdot \mathrm{d}(P, P^k)$ and k (in green).

The above examples serve to illustrate the fact that different polytopes, behave very differently when we try and approximate them using sparse inequalities. We note here that in all our additional experiments, albeit in small dimensions, we have usually found at most three phases as in the previous examples.

1.2 Main Results

1.2.1 Upper Bounds

Surprisingly, it appears that the complicated behavior of $\mathrm{d}(P, P^k)$ as k changes can be described to some extent in closed form. Our first result is nontrivial upper bounds on $\mathrm{d}(P, P^k)$ for general polytopes. This is proven in Section 1.3.

Theorem 1 (Upper Bound on $\mathrm{d}(P, P^k)$). *Let $n \geq 2$. Let $P \subseteq [0,1]^n$ be the convex hull of points $\{p^1, \ldots, p^t\}$. Then*

1. $d(P, P^k) \leq 4 \max \left\{ \frac{n^{1/4}}{\sqrt{k}} \sqrt{8 \max_{i \in [t]} \|p^i\|} \sqrt{\log 4tn}, \frac{8\sqrt{n}}{3k} \log 4tn \right\}$
2. $d(P, P^k) \leq 2\sqrt{n} \left(\frac{n}{k} - 1 \right)$.

Since $\max_{i \in \{1, \ldots, t\}} ||p^i|| \leq \sqrt{n}$ and the first upper bound yields nontrivial values only when $k \geq 8 \log 4tn$, a simpler (although weaker) expression for the first upper bound is $4 \frac{\sqrt{n}}{\sqrt{k}} \sqrt{\log 4tn}$. We make two observations based on Theorem 1.

Consider polytopes with 'few' vertices, say n^q vertices for some constant q. Suppose we decide to use cutting-planes with half sparsity (i.e. $k = \frac{n}{2}$), a reasonable assumption in practice. Then plugging in these values, it is easily verified that $\mathrm{d}(P, P^k) \leq 4\sqrt{2}\sqrt{(q+1)\log n} \approx c\sqrt{\log n}$ for a constant c, which is a significantly small quantity in comparison to $\sqrt{n}$. In other words, *if the number of vertices is small, independent of the location of the vertices, using half sparsity cutting-planes allows us to approximate the integer hull very well.* We believe that as the number of vertices increase, the structure of the polytope becomes more important in determining $\mathrm{d}(P, P^k)$ and Theorem 1 only captures the worst-case scenario. Overall, Theorem 1 presents a theoretical justification for the use of sparse cutting-planes in many cases.

Theorem 1 supports the existence of three phases in the behavior of $\mathrm{d}(P, P^k)$ as k varies: **(Small k)** When $k \leq 16 \log 4tn$ the (simplified) upper bounds are larger than $\sqrt{n}$, indicating that 'no progress' is made in approximating the shape of P (this is seen Examples 2 and 3). **(Medium k)** When $16 \log 4tn \leq k \lesssim n - \sqrt{n \log 4tn}$ the first upper bound in Theorem 1 dominates. **(Large k)** When $k \gtrsim n - \sqrt{n \log 4tn}$ the upper bound $2\sqrt{n}\left(\frac{n}{k} - 1\right)$ dominates. In particular, in this phase, $k \cdot \mathrm{d}(P, P^k) \leq 2n^{3/2} - 2\sqrt{n}k$, i.e., the upper bound times k is a linear function of k. All the examples in Section 1.1 illustrate this behavior.

1.2.2 Lower Bounds

How good is the quality of the upper bound presented in Theorem 1? Let us first consider the second upper bound in Theorem 1. Then observe that for the second example in Section 1.1, this upper bound is tight up to a constant factor for k between the values of $\frac{n}{2}$ and n.

We study lower bounds on $\mathrm{d}(\boldsymbol{P}, \boldsymbol{P}^k)$ for random 0/1 polytopes in Section 1.4 that show that the first upper bound in Theorem 1 is also quite tight.

Theorem 2. *Let $\boldsymbol{X}^1, \boldsymbol{X}^2, \ldots, \boldsymbol{X}^t$ be independent uniformly random points in $\{0,1\}^n$, and let $\boldsymbol{P} = conv(\boldsymbol{X}^1, \boldsymbol{X}^2, \ldots, \boldsymbol{X}^t)$. Then for t and k satisfying $(2k^2 \log n + 2)^2 \leq t \leq e^n$ we have with probability at least $1/4$*

$$d(\boldsymbol{P}, \boldsymbol{P}^k) \geq \min\left\{\frac{\sqrt{n}}{\sqrt{k}} \frac{\sqrt{\log(t/2)}}{78\sqrt{\log n}}, \frac{\sqrt{n}}{8}\right\}\left(\frac{1}{2} - \frac{1}{k^{3/2}}\right) - 3\sqrt{\log t}.$$

Let us compare this lower bound with the simpler expression $4\frac{\sqrt{n}}{\sqrt{k}}\sqrt{\log tn}$ for the first part of the upper bound of Theorem 1. We focus on the case where the minimum in the lower bound is achieved by the first term. Then comparing the leading term $\sqrt{\frac{n}{k}}\frac{\sqrt{\log t}}{2{\cdot}78\sqrt{\log n}}$ in the lower bound with the upper bound, we see that these quantities match up to a factor of $624\frac{\sqrt{\log(tn)}\sqrt{\log n}}{\sqrt{\log t}}$, showing that for many 0/1 polytopes the first upper bound of Theorem 1 is quite tight. We also remark that the in order to simplify the exposition we did not try to optimize constants and lower order terms in our bounds.

The main technical tool for proving this lower bound is a new anticoncentration result for linear combinations $a\boldsymbol{X}$, where the $\boldsymbol{X}_i$'s are independent Bernoulli random variables (Lemma 8). The main difference from standard anticoncentration results is that the latter focus on variation around the standard deviation; in this case, standard tools such as the Berry-Esseen Theorem or the Paley-Zygmund Inequality can be used to obtain constant-probability anticoncentration. However, we need to control the behavior of $a\boldsymbol{X}$ much further from its standard deviation, where we cannot hope to get constant-probability anticoncentration.

1.2.3 Hard Packing Integer Programs

We also study well-known, randomly generated, hard packing integer program instances (see for instance [25]). Given parameters $n, m, M \in \mathbb{N}$, the convex hull of the packing IP is given by $\boldsymbol{P} = \mathrm{conv}(\{x \in \{0,1\}^n : \boldsymbol{A}^j x \leq \frac{\sum_i \boldsymbol{A}_i^j}{2},\ \forall j \in [m]\})$, where the $\boldsymbol{A}_i^j$'s are chosen independently and uniformly in the set $\{0, 1, \ldots, M\}$. Let (n, m, M)-PIP denote the distribution over the generated P's.

The following result shows the limitation of sparse cuts for these instances.

Theorem 3. *Consider $n, m, M \in \mathbb{N}$ such that $n \geq 50$ and $8 \log 8n \leq m \leq n$. Let $\boldsymbol{P}$ be sampled from the distribution (n, m, M)-PIP. Then with probability at least $1/2$,*

$d(\boldsymbol{P}, \boldsymbol{P}^k) \geq \frac{\sqrt{n}}{2}\left(\frac{2}{\max\{\alpha,1\}}(1-\epsilon)^2 - (1+\epsilon')\right)$, *where* $c = k/n$ *and*

$$\frac{1}{\alpha} = \frac{M}{2(M+1)}\left[\frac{n - 2\sqrt{n \log 8m}}{c((2-c)n+1) + 2\sqrt{10cnm}}\right], \qquad \epsilon = \frac{24\sqrt{\log 4n^2 m}}{\sqrt{n}},$$
$$\epsilon' = \frac{3\sqrt{\log 8n}}{\sqrt{m} - 2\sqrt{\log 8n}}.$$

Notice that when m is sufficiently large, and n reasonably larger than m, we have ϵ and ϵ' approximately 0, and the above bound reduces to approximately

$$\frac{\sqrt{n}}{2}\left(\left(\frac{M}{M+1}\right)\left(\frac{n}{k(2-n/k)}\right) - 1\right) \approx \frac{\sqrt{n}}{2}\left(\frac{n}{k(2-n/k)} - 1\right)$$

, which is within a constant factor of the upper bound from Theorem 1. The poor behavior of sparse cuts gives an indication for the hardness of these instances and suggests that denser cuts should be explored in this case.

One interesting feature of this result is that it works directly with the IP formulation, not relying on an explicit linear description of the convex hull.

1.2.4 Sparse Cutting-Planes and Extended Formulations

Let $\text{proj}_x : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ denote the projection operator onto the first n coordinates. We say that a set $Q \subseteq \mathbb{R}^n \times \mathbb{R}^m$ is an *extended formulation* of $P \subseteq \mathbb{R}^n$ if $P = \text{proj}_x(Q)$.

As our final result we remark that using sparse cutting-planes in extended formulations is at least as good as using them in the original polyhedron, and sometime much better. These results are proved in Section 1.6.

Lemma 4. *Consider a polyhedron* $P \subseteq \mathbb{R}^n$ *and an extended formulation* $Q \subseteq \mathbb{R}^n \times \mathbb{R}^m$ *for it. Then* $\text{proj}_x(Q^k) \subseteq (\text{proj}_x(Q))^k = P^k$.

Lemma 5. *Consider* $n \in \mathbb{N}$ *and assume it is a power of 2. Then there is a polytope* $P \subseteq \mathbb{R}^n$ *such that: 1)* $d(P, P^k) = \sqrt{n/2}$ *for all* $k \leq n/2$*; 2) there is an extended formulation* $Q \subseteq \mathbb{R}^n \times \mathbb{R}^{2n-1}$ *of* P *such that* $\text{proj}_x(Q^3) = P$.

1.3 Upper Bound

In this section we prove Theorem 1. In fact we prove the same bound for polytopes in $[-1,1]^n$, which is a slightly stronger result. The following well-known property is crucial for the constructions used in both parts of the theorem.

Observation 3 (Section 2.5.1 of [9])**.** *Consider a compact convex set $S \subseteq \mathbb{R}^n$. Let $\bar{x}$ be a point outside S and let $\bar{y}$ be the closest point to $\bar{x}$ in S. Then setting $a = \bar{x} - \bar{y}$, the inequality $ax \leq a\bar{y}$ is valid for S and cuts $\bar{x}$ off.*

1.3.1 Proof of First Part of Theorem 1

Consider a polytope $P = \text{conv}\{p^1, p^2, \ldots, p^t\}$ in $[-1,1]^n$. Define

$$\lambda^* = \max\left\{\frac{n^{1/4}}{\sqrt{k}}\sqrt{8 \max_i \|p^i\|}\sqrt{\log 4tn}, \frac{8\sqrt{n}}{3k}\log 4tn\right\}.$$

In order to show that $\mathrm{d}(P, P^k)$ is at most $4\lambda^*$ we show that every point at distance more than $4\lambda^*$ from P is cut off by a valid inequality for P^k. Assume until the end of this section that $4\lambda^*$ is at most $\sqrt{n}$, otherwise the result is trivial; in particular, this implies that the second term in the definition of λ^* is at most $\sqrt{n}/4$ and hence $k \geq 8 \log 4tn$.

So let $u \in \mathbb{R}^n$ be a point at distance more than $4\lambda^*$ from P. Let $v \in P$ be the closest point in P to P^k. We can write $u = v + \lambda d$ for some vector d with $\|d\|_2 = 1$ and $\lambda > 4\lambda^*$. From Observation 3, inequality $dx \leq dv$ is valid for P, so in particular $dp^i \leq dv$ for all $i \in [t]$; in addition, it that this inequality cuts off u: $du = dv + \lambda > dv$. The idea is to use this extra slack factor λ in the previous equation to show we can 'sparsify' the inequality $dx \leq dv$ while maintaining separation of P and u. It then suffices to prove the following lemma.

Lemma 6. *There is a k-sparse vector $\tilde{d} \in \mathbb{R}^n$ such that $\tilde{d}p^i \leq \tilde{d}v + \frac{\lambda}{2}$ for all $i \in [t]$, and $\tilde{d}u > \tilde{d}v + \frac{\lambda}{2}$.*

To prove the lemma we construct a random vector $\tilde{\boldsymbol{D}} \in \mathbb{R}^n$ which, with non-zero probability, is k-sparse and satisfies the two other requirements of the lemma. Let $\alpha = \frac{k}{2\sqrt{n}}$. Define $\tilde{\boldsymbol{D}}$ as the random vector with independent coordinates, where $\tilde{\boldsymbol{D}}_i$ is defined as follows: if $\alpha|d_i| \geq 1$, then $\tilde{\boldsymbol{D}}_i = d_i$ with probability 1; if $\alpha|d_i| < 1$, then $\tilde{\boldsymbol{D}}_i$ takes value $\text{sign}(d_i)/\alpha$ with probability $\alpha|d_i|$ and takes value 0 with probability $1 - \alpha|d_i|$. (For convenience we define $\text{sign}(0) = 1$.)

The next observation follows directly from the definition of $\tilde{\boldsymbol{D}}$.

Observation 4. *For every vector $a \in \mathbb{R}^n$ the following hold:*

1. $\mathbb{E}[\tilde{\boldsymbol{D}}a] = da$

2. $Var(\tilde{\boldsymbol{D}}a) \leq \frac{1}{\alpha}\sum_{i\in[n]} a_i^2|d_i|$

3. $|\tilde{\boldsymbol{D}}_i a_i - \mathbb{E}[\tilde{\boldsymbol{D}}_i a_i]| \leq \frac{|a_i|}{\alpha}$.

Claim 1. *With probability at least $1 - 1/4n$, $\tilde{\boldsymbol{D}}$ is k-sparse.*

Proof. Construct the vector $a \in \mathbb{R}^n$ as follows: if $\alpha|d_i| \geq 1$ then $a_i = 1/d_i$, and if $\alpha|d_i| < 1$ then $a_i = \alpha/\text{sign}(d_i)$. Notice that $\tilde{\boldsymbol{D}}a$ equals the number of non-zero coordinates of $\tilde{\boldsymbol{D}}$ and $\mathbb{E}[\tilde{\boldsymbol{D}}a] \leq \alpha\|d\|_1 \leq k/2$. Also, from Observation 4 we have

$$\text{Var}(\tilde{\boldsymbol{D}}a) \leq \frac{1}{\alpha}\sum_{i\in[n]} a_i^2|d_i| \leq \alpha\|d\|_1 \leq \frac{k}{2}.$$

Then using Bernstein's inequality (Section A.1 of the appendix) we obtain

$$\Pr(\tilde{\boldsymbol{D}}a \geq k) \leq \exp\left(-\min\left\{\frac{k^2}{8k}, \frac{3k}{8}\right\}\right) \leq \frac{1}{4n},$$

where the last inequality uses our assumption that $k \geq 8\log 4tn$. $\square$

We now show that property 1 required by Lemma 6 holds for $\tilde{\boldsymbol{D}}$ with high probability.

Claim 2. $\Pr(\max_{i\in[t]}[\tilde{\boldsymbol{D}}(p^i - v) - d(p^i - v)] > 2\lambda^*) \leq 1/4n.$

Proof. Define the centered random variable $\boldsymbol{Z} = \tilde{\boldsymbol{D}} - d$. To make the analysis cleaner, notice that $\max_{i \in [t]} \boldsymbol{Z}(p^i - v) \le 2 \max_{i \in [t]} |\boldsymbol{Z} p^i|$; this is because $\max_{i \in [t]} \boldsymbol{Z}(p^i - v) \le \max_{i \in [t]} |\boldsymbol{Z} p^i| + |\boldsymbol{Z} v|$, and because for all $a \in \mathbb{R}^n$ we have $|av| \le \max_{p \in P} |ap| = \max_{i \in [t]} |ap^i|$ (since $v \in P$).

Therefore our goal is to upper bound the probability that the process $\max_{i \in [t]} |\boldsymbol{Z} p^i|$ is larger then λ^*. Fix $i \in [t]$. By Bernstein's inequality,

$$\Pr(|\boldsymbol{Z} p^i| \ge \lambda^*) \le \exp\left(- \min\left\{ \frac{(\lambda^*)^2}{4\mathrm{Var}(|\boldsymbol{Z} p^i|)}, \frac{3\lambda^*}{4M} \right\} \right), \tag{1}$$

where M is an upper bound on $\max_j |\boldsymbol{Z}_j p^i_j|$.

To bound the terms in the right-hand side, from Observation 4 we have

$$\mathrm{Var}(\boldsymbol{Z} p^i) = \mathrm{Var}(\tilde{\boldsymbol{D}} p^i) \le \frac{1}{\alpha} \sum_j (p^i_j)^2 |d_j| \le \frac{1}{\alpha} \sum_j p^i_j |d_j| \le \frac{1}{\alpha} \|p^i\| \|d\| = \frac{1}{\alpha} \|p^i\|,$$

where the second inequality follows from the fact $p^i \in [0,1]^n$, and the third inequality follows from the Cauchy-Schwarz inequality. Moreover, it is not difficulty to see that for every random variable $\boldsymbol{W}$, $\mathrm{Var}(|\boldsymbol{W}|) \le \mathrm{Var}(\boldsymbol{W})$. Using the first term in the definition of λ^*, we then have

$$\frac{(\lambda^*)^2}{\mathrm{Var}(|\boldsymbol{Z} p^i|)} \ge 4 \log 4tn.$$

In addition, for every coordinate j we have $|\boldsymbol{Z}_j p^i_j| = |\tilde{\boldsymbol{D}}_j p^i_j - \mathbb{E}[\tilde{\boldsymbol{D}}_j p^i_j]| \le 1/\alpha$, where the inequality follows from Observation 4. Then we can set $M = 1/\alpha$ and using the second term in the definition of λ^* we get $\frac{\lambda^*}{M} \ge \frac{4}{3} \log 4tn$. Therefore, replacing these bounds in inequality (1) gives $\Pr(|\boldsymbol{Z} p^i| \ge \lambda^*) \le \frac{1}{4tn}$.

Taking a union bound over all $i \in [t]$ gives that $\Pr(\max_{i \in [t]} |\boldsymbol{Z} p^i| \ge \lambda^*) \le 1/4n$. This concludes the proof of the claim. ☐

Claim 3. $\Pr(\tilde{\boldsymbol{D}}(u - v) \le \lambda/2) \le 1 - 1/(2n-1)$.

Proof. Recall $u - v = \lambda d$, hence it is equivalent to bound $\Pr(\tilde{\boldsymbol{D}} d \le 1/2)$. First, $\mathbb{E}[\tilde{\boldsymbol{D}} d] = dd = 1$. Also, from Observation 4 we have $\tilde{\boldsymbol{D}} d \le |\tilde{\boldsymbol{D}} d - dd| + |dd| \le$

$\frac{1}{\alpha}\sum_i |d_i|+1 \le \frac{2n}{k}+1 \le n$, where the last inequality uses the assumption $k \ge 8\log 4tn$. Then employing Markov's inequality to the non-negative random variable $n - \tilde{\boldsymbol{D}}d$, we get $\Pr(\tilde{\boldsymbol{D}}d \le 1/2) \le 1 - \frac{1}{2n-1}$. This concludes the proof. □

Proof of Lemma 6. Employ the previous three claims and union bound to find a realization of $\tilde{\boldsymbol{D}}$ that is k-sparse and satisfies requirements 1 and 2 of the lemma.

This concludes the proof of the first part of Theorem 1.

Observation 5. *Notice that in the above proof λ^* is set by Claim 2, and need to be essentially $\mathbb{E}[\max_{i\in[t]}(\tilde{\boldsymbol{D}}-d)p^i]$. There is a vast literature on bounds on the supremum of stochastic processes (see for instance [27]), and improved bounds for structured P's are possible (for instance, via the* generic chaining *method).*

1.3.2 Proof of Second Part of Theorem 1

The main tool for proving this upper bound is the following lemma, which shows that when P is 'simple', and we have a stronger control over the distance of a point $\bar{x}$ to P, then there is a k-sparse inequality that cuts $\bar{x}$ off.

Lemma 7. *Consider a hyperplane $H = \{x \in \mathbb{R}^n : ax \le b\}$ and let $P = H \cap [-1,1]^n$. Let $\bar{x} \in [-1,1]^n$ be such that $d(\bar{x}, H) > 2\sqrt{n}(\frac{n}{k}-1)$. Then $\bar{x} \notin P^k$.*

Proof. Assume without loss of generality that $\|a\|_2 = 1$. Let $\bar{y}$ be the point in H closes to $\bar{x}$, and notice that $\bar{x} = \bar{y} + \lambda a$ where $\lambda > \sqrt{n}(\frac{n}{k}-1)$.

For any set $I \in \binom{[n]}{k}$, the inequality $\sum_{i\in I} a_i x_i \le b + \sum_{i\notin I: a_i \ge 0} a_i - \sum_{i \notin I: a_i<0} a_i$ is valid for P; since it is k-sparse, it is also valid for P^k. Averaging out this inequality over all $I \in \binom{[n]}{k}$, we get that the following is valid for P^k:

$$\frac{k}{n}ax \le b + \left(1-\frac{k}{n}\right)\left(\sum_{i:a_i\ge 0} a_i - \sum_{i:a_i<0} a_i\right) \equiv ax \le b + \left(\frac{n}{k}-1\right)(b + \|a\|_1).$$

We claim that $\bar{x}$ violates this inequality. First notice that $a\bar{x} = a\bar{y} + \lambda = b + \lambda > b + 2\sqrt{n}\left(\frac{n}{k}-1\right)$, hence it suffices to show $b + \|a\|_1 \le 2\sqrt{n}$. Our assumption on $\bar{x}$ implies that $P \neq [-1,1]^n$, and hence $b < \max_{x \in [-1,1]} ax = \|a\|_1$; this gives $b + \|a\|_1 \leq 2\|a\|_1 \leq 2\sqrt{n}\|a\|_2 = 2\sqrt{n}$, thus concluding the proof. $\square$

To prove the second part of Theorem 1 consider a point $\bar{x}$ of distance greater than $2\sqrt{n}(\frac{n}{k} - 1)$ from P; we show $\bar{x} \notin P^k$. Let $\bar{y}$ be the closest point to $\bar{x}$ in P. Let $a = \bar{x} - \bar{y}$. From Observation 3 we have that $ax \leq a\bar{y}$ is valid for P. Define $H' = \{x \in \mathbb{R}^n : ax \leq a\bar{y}\}$ and $P' = H' \cap [-1,1]^n$. Notice that $\mathrm{d}(\bar{x}, H') = \mathrm{d}(\bar{x}, \bar{y}) > 2\sqrt{n}(\frac{n}{k} - 1)$. Then Lemma 7 guarantees that $\bar{x}$ does not belong to P'^k. But $P \subseteq P'$, so by monotonicity of the k-sparse closure we have $P^k \subseteq P'^k$; this shows that $\bar{x} \notin P^k$, thus concluding the proof.

1.4 Lower Bound

In this section we prove Theorem 2. The proof is based on the 'bad' polytope of Example 2. For a random polytope $\boldsymbol{Q}$ in $\mathbb{R}^n$, it is useful to think of each of its (random) faces from the perspective of supporting hyperplanes: for a fixed direction $d \in \mathbb{R}^n$, we have the valid inequality $dx \leq \boldsymbol{d}_0$, where $\boldsymbol{d}_0 = \max_{q \in \boldsymbol{Q}} dq$.

The idea of the proof is then to proceed in two steps. First, for a uniformly random 0/1 polytope $\boldsymbol{P}$, we show that with good probability the faces $dx \leq \boldsymbol{d}_0$ for $\boldsymbol{P}^k$ have $\boldsymbol{d}_0$ being large, namely $\boldsymbol{d}_0 \gtrsim \left(\frac{1}{2} + \frac{\sqrt{\log t}}{\sqrt{k}}\right) \sum_i d_i$, forced by some point $p \in \boldsymbol{P}$ with large dp; therefore, with good probability the point $\bar{p} \approx (\frac{1}{2} + \frac{\sqrt{\log t}}{\sqrt{k}})e$ belongs to $\boldsymbol{P}^k$. In the second step, we show that with good probability the distance from $\bar{p}$ to $\boldsymbol{P}$ is at least $\approx \sqrt{\frac{n}{k}}\sqrt{\log t}$, by showing that the inequality $\sum_i x_i \lesssim \frac{n}{2} + \sqrt{n}$ is valid for $\boldsymbol{P}$.

We now proceed with the proof. Consider the random set $\boldsymbol{\mathcal{X}} = \{\boldsymbol{X}^1, \boldsymbol{X}^2, \ldots, \boldsymbol{X}^t\}$ where the $\boldsymbol{X}^i$'s are independent uniform random points in $\{0,1\}^n$, and define the random 0/1 polytope $\boldsymbol{P} = \mathrm{conv}(\boldsymbol{\mathcal{X}})$. To formalize the preceding discussion, we say that a (deterministic) 0/1 polytope in $\mathbb{R}^n$ is α-*tough* if for every facet $dx \leq d_0$ of its k-sparse closure we have $d_0 \geq \frac{\sum_i d_i}{2} + \frac{\alpha}{2\sqrt{k}}(1 - \frac{1}{k^2})\|d\|_1 - \|d\|_\infty/2k^2$, for every $k \in [n]$.

The main element of the lower bound is the following anticoncentration result; in our setting, the idea is that it gives that for every (k-sparse) direction $d \in \mathbb{R}^n$, with good probability we will have a point p in $\boldsymbol{P}^k$ (in fact in $\boldsymbol{P}$) with large dp.

Lemma 8. *Let $\boldsymbol{Z}_1, \boldsymbol{Z}_2, \ldots, \boldsymbol{Z}_n$ be independent random variables with $\boldsymbol{Z}_i$ taking value 0 with probability 1/2 and value 1 with probability 1/2. Then for every $a \in [-1,1]^n$ and $\alpha \in [0, \frac{\sqrt{n}}{8}]$,*

$$\Pr\left(a\boldsymbol{Z} \geq \mathbb{E}[a\boldsymbol{Z}] + \frac{\alpha}{2\sqrt{n}}\left(1 - \frac{1}{n^2}\right)\|a\|_1 - \frac{1}{2n^2}\right) \geq \left(e^{-50\alpha^2} - e^{-100\alpha^2}\right)^{60\log n}.$$

The proof of this lemma is reasonably simple and proceeds by grouping the random variables with similar a_i's and then applies known anticoncentration to each of these groups; this proof is presented in Section A.3 of the appendix.

In order to effectively apply this anticoncentration to *all* valid inequalities/directions of $\boldsymbol{P}^k$, we need some additional control. Define $\mathcal{D}$ as the set of all integral vectors $\ell \in \mathbb{R}^n$ that are k-sparse and satisfy $\|\ell\|_\infty \leq (k+1)^{(k+1)/2}$.

Lemma 9. *Let $Q \subseteq \mathbb{R}^n$ be a 0/1 polytope. Then for every $k \in [n]$, there is a subset $\mathcal{D}' \subseteq \mathcal{D}$ such that $Q^k = \{x : dx \leq \max_{y \in Q^k} dy,\ d \in \mathcal{D}'\}$.*

This lemma follows directly from applying Corollary 26 in [40] to each term $Q + \mathbb{R}^{\bar{I}}$ in the definition of Q^k from Section 1.1.1.

Employing this lemma to each scenario, we get that all the directions of facets of $\boldsymbol{P}^k$ come from the set $\mathcal{D}$. This allow us to analyze the probability that $\boldsymbol{P}$ is α-tough.

Lemma 10. *If $1 \leq \alpha^2 \leq \min\left\{\frac{\log(t/2)}{6000\log n}, \frac{k}{64}\right\}$ and $k \leq n-1$, then $\boldsymbol{P}$ is α-tough with probability at least $1/2$.*

Proof. Let $\mathcal{E}$ be the event that for all $d \in \mathcal{D}$ we have $\max_{i \in [t]} d\boldsymbol{X}^i \geq \frac{1}{2}\sum_j d_j + \frac{\alpha}{2\sqrt{k}}(1 - \frac{1}{k^2})\|d\|_1 - \|d\|_\infty/2k^2$. Because of Lemma 9, whenever $\mathcal{E}$ holds we have that $\boldsymbol{P}$ is α-tough and thus it suffices to show $\Pr(\mathcal{E}) \geq 1/2$.

Fix $d \in \mathcal{D}$. Since d is k-sparse, we can apply Lemma 8 to $d/\|d\|_\infty$ restricted to the coordinates in its support to obtain that

$$\Pr\left(d\boldsymbol{X}^i \geq \frac{\sum_i d_i}{2} + \frac{\alpha}{2\sqrt{k}}\left(1 - \frac{1}{k^2}\right)\|d\|_1 - \frac{\|d\|_\infty}{2k^2}\right) \geq \left(e^{-50\alpha^2} - e^{-100\alpha^2}\right)^{60\log n}$$
$$\geq e^{-100\alpha^2 \cdot 60\log n} \geq \frac{1}{t^{1/2}},$$

where the second inequality follows from our lower bound on α and the last inequality follows from our upper bound on α. By independence of the $\boldsymbol{X}^i$'s,

$$\Pr\left(\max_{i\in[t]} d\boldsymbol{X}^i < \frac{\sum_i d_i}{2} + \frac{\alpha}{2\sqrt{k}}\left(1 - \frac{1}{k^2}\right)\|d\|_1 - \frac{\|d\|_\infty}{2k^2}\right) \leq \left(1 - \frac{1}{t^{1/2}}\right)^t \leq e^{-t^{1/2}},$$

where the second inequality follows from the fact that $(1-x) \leq e^{-x}$ for all x.

Finally notice that $|\mathcal{D}| = \binom{n}{k}(k+1)^{(k+1)^2/2}$ and that, by our assumption on the size of t and $k \leq n-1$, $e^{-t^{1/2}} \leq (1/2)|\mathcal{D}|$. Therefore, taking a union bound over all $d \in \mathcal{D}$ of the previous displayed inequality gives $\Pr(\mathcal{E}) \geq 1/2$, concluding the proof of the lemma. $\square$

The next lemma takes care of the second step of the argument.

Lemma 11. *With probability at least* $3/4$*, the inequality* $\sum_j x_j \leq \frac{n}{2} + 3\sqrt{n\log t}$ *is valid for* $\boldsymbol{P}$*.*

Proof. Fix an $i \in [t]$. Since $\mathrm{Var}(\boldsymbol{X}^i) = n/4$, we have from Bernstein's inequality

$$\Pr\left(\sum_j \boldsymbol{X}^i_j \geq \frac{n}{2} + 3\sqrt{n\log t}\right) \leq \exp\left(-\min\left\{9\log t, \frac{9\sqrt{n\log t}}{4}\right\}\right) \leq e^{-\frac{9\log t}{4}} \leq \frac{1}{4t},$$

where the second inequality follows from the fact that $\log t \leq n$, and the last inequality uses the fact that $t \geq 4$. Taking a union bound over all $i \in [t]$ gives

$$\Pr\left(\bigvee_{i\in[t]}\left(\sum_j \boldsymbol{X}^i_j \geq \frac{n}{2} + 3\sqrt{n\log t}\right)\right) \leq \frac{1}{4},$$

Finally, notice that an inequality $dx \leq d_0$ is valid for $\boldsymbol{P}$ iff it is valid for all $\boldsymbol{X}^i$. This concludes the proof. $\square$

Lemma 12. *Suppose that the polytope Q is α-tough for $\alpha \geq 1$ and that the inequality $\sum_i x_i \leq \frac{n}{2} + 3\sqrt{n \log t}$ is valid for Q. Then we have $d(Q, Q^k) \geq \sqrt{n}\left(\frac{\alpha}{2\sqrt{k}} - \frac{\alpha}{k^2} - \frac{3\sqrt{\log t}}{\sqrt{n}}\right)$.*

Proof. We first show that the point $\bar{q} = (\frac{1}{2} + \frac{\alpha}{2\sqrt{k}} - \frac{\alpha}{k^2})e$ belongs to P. Let $dx \leq d_0$ be facet for P. Then we have

$$\begin{aligned} d\bar{q} &= \frac{\sum_i d_i}{2} + \alpha\left(\frac{1}{2\sqrt{k}} - \frac{1}{k^2}\right)\sum_i d_i \leq \frac{\sum_i d_i}{2} + \alpha\left(\frac{1}{2\sqrt{k}} - \frac{1}{k^2}\right)\|d\|_1 \\ &\leq \frac{\sum_i d_i}{2} + \alpha\left(\frac{1}{2\sqrt{k}} - \frac{1}{2k^2}\right)\|d\|_1 - \frac{\|d\|_\infty}{2k^2}, \end{aligned}$$

where the first inequality uses the fact that $\frac{1}{2\sqrt{k}} - \frac{1}{k^2} \geq 0$ and the second inequality uses $\alpha \geq 1$ and $\|d\|_1 \geq \|d\|_\infty$. Since Q is α-tough it follows that $\bar{q}$ satisfies $dx \leq d_0$; since this holds for all facets of Q, we have $\bar{q} \in Q$.

Now define the halfspace $H = \{x : \sum_i x_i \leq \frac{n}{2} + 3\sqrt{n \log t}\}$. By assumption $Q \subseteq H$, and hence $\mathrm{d}(Q, Q^k) \geq \mathrm{d}(H, Q^k)$. But it is easy to see that the point in H closest to $\bar{q}$ is the point $\tilde{q} = (\frac{1}{2} + \frac{3\sqrt{\log t}}{\sqrt{n}})e$. This gives that $\mathrm{d}(Q, Q^k) \geq \mathrm{d}(H, Q^k) \geq \mathrm{d}(\bar{q}, \tilde{q}) \geq \sqrt{n}\left(\frac{\alpha}{2\sqrt{k}} - \frac{\alpha}{k^2} - \frac{3\sqrt{\log t}}{\sqrt{n}}\right)$. This concludes the proof. $\square$

We now conclude the proof of Theorem 2. Set $\bar{\alpha}^2 = \min\left\{\frac{\log(t/2)}{6000 \log n}, \frac{k}{64}\right\}$. Taking union bound over Lemmas 10 and 11, with probability at least 1/4, $\boldsymbol{P}$ is $\bar{\alpha}$-tough and the inequality $\sum_i x_i \leq \frac{n}{2} + 3\sqrt{n \log t}$ is valid for it. Then from Lemma 12 we get that with probability at least 1/4, $\mathrm{d}(\boldsymbol{P}, \boldsymbol{P}^k) \geq \sqrt{n}\left(\frac{\bar{\alpha}}{2\sqrt{k}} - \frac{\bar{\alpha}}{k^2} - \frac{3\sqrt{\log t}}{\sqrt{n}}\right)$, and the result follows by plugging in the value of $\bar{\alpha}$.

1.5 Hard Packing Integer Programs

In this section we prove Theorem 3. With overload in notation, we use $\binom{[n]}{k}$ to denote the set of vectors in $\{0, 1\}^n$ with exactly k 1's.

Let $\boldsymbol{P}$ be a random polytope sampled from the distribution (n, m, M)-PIP and consider the corresponding random vectors $\boldsymbol{A}^j$'s. The idea of the proof is to show that with constant probability $\boldsymbol{P}$ behaves like Example 2, by showing that the cut

$\sum_i x_i \lesssim \frac{n}{2}$ is valid for it and that $\boldsymbol{P}$ approximately contains 0/1 points with many 1's. Then we show that this 'approximate containment' implies that a point with a lot of mass (say, $\approx (1, 1, \ldots, 1)$ for $k \leq n/2$) belongs to the k-sparse closure $\boldsymbol{P}^k$; since such point is far from hyperplane $\sum_i x_i \lesssim \frac{n}{2}$, it is also far from $\boldsymbol{P}$ and hence we get a lower bound on $\mathrm{d}(\boldsymbol{P}, \boldsymbol{P}^k)$.

The first part of the argument is a straightforward application of Bernstein's inequality and union bound; its proof is presented in Section A.4 of the appendix.

Lemma 13. *With probability at least* $1 - \frac{1}{4}$ *the cut* $(1 - \frac{2\sqrt{\log 8n}}{\sqrt{m}}) \sum_i x_i \leq \frac{n}{2} + \frac{\sqrt{n \log 8}}{\sqrt{m}}$ *is valid for* $\boldsymbol{P}$.

The other steps in the argument are more involved.

1.5.1 Approximate Containment of Points with Many 1's

First we control the right-hand side of the constraints $\boldsymbol{A}^j x \leq \frac{\sum_i \boldsymbol{A}_i^j}{2}$ that define $\boldsymbol{P}$, by showing that they are roughly $\frac{nM}{2}$; this is again a straightforward application of Bernstein's inequality and is also deferred to Section A.4 of the appendix.

Lemma 14. *With probability at least* $1 - \frac{1}{8}$ *we have* $|\sum_{i=1}^n \boldsymbol{A}_i^j - \frac{nM}{2}| \leq M\sqrt{n \log 8m}$ *for all* $j \in [m]$.

Recall that we defined $c = \frac{k}{n}$. Now we show that with constant probability, *all* points $\bar{x} \in \{0,1\}^n$ with cn 1's satisfy $\boldsymbol{A}^j \lesssim \frac{nM}{2}$ for all $j \in [m]$, and hence they approximately belong to $\boldsymbol{P}$. The argument is cleaner is the random variables $\boldsymbol{A}_i^j$ were uniformly distributed in the *continuous* interval $[0, M]$, instead of on the discrete set $\{0, \ldots, M\}$; this is because in the former we can leverage the knowledge of the *order statistics* of continuous uniform variables. Our next lemma then essentially handles this continuous case.

Lemma 15. *Let* $\boldsymbol{U} \in \mathbb{R}^n$ *be a random variable where each coordinate* $\boldsymbol{U}_i$ *is independently drawn uniformly from* $[0,1]$. *Then with probability at least* $1 - 1/8m$ *we have* $\boldsymbol{U}\bar{x} \leq \frac{c(2n - cn + 1)}{2} + \sqrt{10cnm}$ *for all vectors* $\bar{x} \in \binom{n}{cn}$.

Proof. Let $\boldsymbol{U}_{(i)}$ be the ith order statistics of $\boldsymbol{U}_1, \boldsymbol{U}_2, \ldots, \boldsymbol{U}_n$ (i.e. in each scenario $U_{(i)}$ equals the ith smallest value among $\boldsymbol{U}_1, \boldsymbol{U}_2, \ldots, \boldsymbol{U}_n$ in that scenario). Notice that $\max_{\bar{x} \in \binom{n}{cn}} \boldsymbol{U}\bar{x} = \boldsymbol{U}_{(n)} + \ldots + \boldsymbol{U}_{(n-cn+1)}$, and hence is it equivalent to show that

$$\Pr\left(\boldsymbol{U}_{(n)} + \ldots + \boldsymbol{U}_{(n-cn+1)} > \frac{c(2n - cn + 1)}{2} + \sqrt{10cnm}\right) \leq \frac{1}{8m}.$$

We use $\boldsymbol{Z} \triangleq \boldsymbol{U}_{(n)} + \ldots + \boldsymbol{U}_{(n-cn+1)}$ to simplify the notation.

It is known that $\mathbb{E}[\boldsymbol{U}_{(i)}] = \frac{i}{n+1}$ and $\mathrm{Cov}(\boldsymbol{U}_{(i)}, \boldsymbol{U}_{(j)}) = \frac{i(n+1-j)}{(n+1)^2(n+2)} \leq \frac{1}{n}$ [14]. Also, since $\boldsymbol{U}_{(i)}$ lies in $[0,1]$, we have $\mathrm{Var}(\boldsymbol{U}_{(i)}) \leq 1/4$. Using this information, we get $\mathbb{E}[\boldsymbol{Z}] = \frac{(2n-cn+1)cn}{2(n+1)} \leq \frac{c(2n-cn+1)}{2}$ and

$$\mathrm{Var}(\boldsymbol{Z}) \leq \frac{cn}{4} + \frac{(cn)^2}{n} \leq \frac{5cn}{4},$$

where the last inequality follows from the fact $c \leq 1$. Then applying Chebyshev's inequality [27], we get

$$\Pr\left(\boldsymbol{Z} \geq \frac{c(2n - cn + 1)}{2} + \sqrt{10cnm}\right) \leq \frac{\mathrm{Var}(\boldsymbol{Z})}{10cnm} \leq \frac{1}{8m}.$$

This concludes the proof. □

Now we translate this proof from the continuous to the discrete setting.

Lemma 16. *With probability at least* $1 - \frac{1}{8}$ *we have*

$$\boldsymbol{A}^j \bar{x} \leq \frac{(M+1)c(2n - cn + 1)}{2} + (M+1)\sqrt{10cnm}, \qquad \forall j \in [m], \forall \bar{x} \in \binom{[n]}{cn}.$$

Proof. For each $j \in [m]$, let $\boldsymbol{U}_1^j, \boldsymbol{U}_2^j, \ldots, \boldsymbol{U}_n^j$ be independent and uniformly distributed in $[0,1]$. Define $\boldsymbol{Y}_i^j \triangleq \lfloor (M+1)\boldsymbol{U}_i^j \rfloor$. Notice that the random variables $(\boldsymbol{Y}_i^j)_{i,j}$ have the same distribution as $(\boldsymbol{A}_i^j)_{i,j}$. So it suffices to prove the lemma for the variables $\boldsymbol{Y}_i^j$'s.

Fix $j \in [m]$. For any $\bar{x} \in \{0,1\}^n$ we have $\boldsymbol{Y}^j \bar{x} \leq (M+1)\boldsymbol{U}\bar{x}$. Therefore, from Lemma 15 we get

$$\Pr\left(\bigvee_{\bar{x} \in \binom{n}{cn}} \left(\boldsymbol{Y}^j \bar{x} > \frac{(M+1)c(2n - cn + 1)}{2} + (M+1)\sqrt{10cnm}\right)\right) \leq \frac{1}{8m}.$$

Taking a union bound of this last expression over all $j \in [m]$ concludes the proof of the lemma. □

1.5.2 From Approximate to Actual Containment

From the previous section we get with constant probability, points $\bar{x} \in \{0,1\}^n$ with cn 1's approximately belong to $\boldsymbol{P}$; thus, scaling them by a small factor, shows that these points belong to the *LP relaxation* of $\boldsymbol{P}$. Our goal is to strengthen this result to show that a small (although slightly larger) scaling of these point actually bring the to the integer hull $\boldsymbol{P}$ *itself*.

The next lemma shows that this is in fact possible.

Lemma 17. *Consider a 0/1 polytope $Q = conv(\{x \in \{0,1\}^n : a^j x \leq b_j,\ j = 1,2,\ldots,m\})$ where $n \geq 20$, $m \leq n$, $a_i^j \in [0,M]$ for all i,j, and $b_j \geq \frac{nM}{12}$ for all i. Consider $1 < \alpha \leq 2\sqrt{n}$ and let $\bar{x} \in \{0,1\}^n$ be such that for all j, $a^j\bar{x} \leq \alpha b_j$. Then the point $\frac{1}{\alpha}(1-\epsilon)^2\bar{x}$ belongs to Q as long as $\frac{12\sqrt{\log 4n^2 m}}{\sqrt{n}} \leq \epsilon \leq \frac{1}{2}$.*

For the remainder of the section we prove this lemma. The idea is that we can select a subset of $\approx 1 - 1/\alpha$ coordinates and change $\bar{x}$ to 0 in these coordinates to obtain a feasible solution in Q; repeating this for many sets of coordinates and taking an average of the feasible points obtained will give the result.

To make this precise, let $p = \frac{1}{\alpha}(1-\epsilon)$. For $w \in [n^2]$ define the independent random variables $\boldsymbol{X}_1^w, \boldsymbol{X}_2^w, \ldots, \boldsymbol{X}_n^w$ taking values in $\{0,1\}$ such that $\mathbb{E}[\boldsymbol{X}_i^w] = p\bar{x}_i$ (i.e. if $\bar{x}_i = 1$, then keep it at 1 with probability p, otherwise flip it to 0; if $\bar{x}_i = 0$, then keep it at 0).

Claim 4. *With probability at least 3/4 all points $\boldsymbol{X}^w$ belong to Q.*

Proof. Notice $\mathbb{E}[a^j\boldsymbol{X}^w] = (1-\epsilon)b_j$. Also, from our upper bound on a^j, we have $\mathrm{Var}(a^j\boldsymbol{X}^w) \leq \frac{M^2 n}{4}$. Employing Bernstein's inequality,

$$\Pr(a^j\boldsymbol{X}^w > b_j) \leq \exp\left(-\min\left\{\frac{\epsilon^2 b_j^2}{M^2 n}, \frac{3\epsilon b_j}{4M}\right\}\right) \leq \frac{1}{4n^2 m},$$

where the second inequality uses the assumed lower bounds on b_j and ϵ, and the fact that $\frac{4 \cdot 12 \log 4n^2 m}{3n} \leq \frac{12\sqrt{\log 4n^2 m}}{\sqrt{n}}$ due to our bounds on n and m. The claim follows by taking a union bound over all j and w. $\square$

Let $\boldsymbol{Z} = \frac{1}{n^2} \sum_w \boldsymbol{X}^w$ be the random point that is the average of the $\boldsymbol{X}^w$'s.

Claim 5. *With probability at least* $3/4$, $\boldsymbol{Z}_i \geq \frac{1}{\alpha}(1-\epsilon)^2 \bar{x}_i$ *for all* i.

Proof. Since $\bar{x} \in \{0,1\}^n$, it suffices to consider indices i such that $\bar{x}_i = 1$. Fix such an i. We have $\mathbb{E}[n^2 \boldsymbol{Z}_i] = pn^2$ and $\mathrm{Var}(n^2 \boldsymbol{Z}_i) \leq \frac{n^2}{4}$. Then from Bernstein's inequality

$$\begin{aligned} \Pr\left(\boldsymbol{Z}_i < \frac{1}{\alpha}(1-\epsilon)^2 \bar{x}_i\right) &= \Pr(n^2 \boldsymbol{Z}_i < \mathbb{E}[n^2 \boldsymbol{Z}_i](1-\epsilon)) \\ &\leq \exp\left(-\min\left\{n^2(\epsilon p)^2, \frac{3n^2 \epsilon p}{4}\right\}\right) \leq \frac{1}{4n}, \end{aligned}$$

where the last inequality uses the lower bound on ϵ, the fact that $n \geq 50$, and the fact that $p \geq 1/2\alpha \geq 1/4\sqrt{n}$. The claim follows from taking a union bound over all i such that $\bar{x}_i = 1$. $\square$

Taking a union bound over the above two claims we get that there is a realization $\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^{n^2}$ of the random vectors $\boldsymbol{X}^1, \boldsymbol{X}^2, \ldots, \boldsymbol{X}^{n^2}$ such that (let $\tilde{z} = \frac{1}{n^2} \sum_w \tilde{x}^w$): (i) All $\tilde{x}^w$ belong to Q, and hence so does their convex combination $\tilde{z}$; (ii) $\tilde{z} \geq \frac{1}{\alpha}(1-\epsilon)^2 \bar{x}$. Since Q is of packing-type, it follows that the point $\frac{1}{\alpha}(1-\epsilon)^2 \bar{x}$ belongs to Q, concluding the proof of Lemma 17.

1.5.3 Proof of Theorem 3

Now we put together the previous sections to conclude the proof of Theorem 3. Recall the definitions of α, ϵ and ϵ' from the statement of the theorem. Let $\mathcal{E}$ be the event that Lemmas 14, 13 and 16 hold; notice that $\Pr(\mathcal{E}) \geq 1/2$. For the rest of the proof we fix a $\boldsymbol{P}$ (and the associated $\boldsymbol{A}^j$'s) where $\mathcal{E}$ holds and prove a lower bound on $\mathrm{d}(\boldsymbol{P}, \boldsymbol{P}^k)$.

Consider a set $I \in \binom{[n]}{cn}$ and let $\bar{x}$ be the incidence vector of I (i.e. $\bar{x}_i = 1$ if $i \in I$ and $\bar{x}_i = 0$ if $i \notin I$). Since the bounds from Lemmas 14 and 16 hold for our $\boldsymbol{P}$, straightforward calculations show that $\boldsymbol{A}^j \bar{x} \leq \alpha \frac{1}{2} \sum_i \boldsymbol{A}^j_i$ for all $j \in [m]$. Therefore, from Lemma 17 we have that the point $\frac{1}{\max\{\alpha,1\}}(1-\epsilon)^2 \bar{x}$ belongs to $\boldsymbol{P}$. This means that the point $\tilde{x} = \frac{1}{\max\{\alpha,1\}}(1-\epsilon)^2 e$ belongs to $\boldsymbol{P} + \mathbb{R}^{\bar{I}}$ (see Section 1.1.1). Since this holds for every $I \in \binom{[n]}{cn}$, we have $\tilde{x} \in \boldsymbol{P}^k$.

Let $\tilde{\boldsymbol{y}}$ be the point in $\boldsymbol{P}$ closest to $\tilde{x}$. Let $a = (1 - \frac{2\sqrt{\log 8n}}{\sqrt{m}})$ and $b = \frac{n}{2} + \sqrt{n \log 8m}$, so that the cut in Lemma 13 is given by $aex \leq b$. From Cauchy-Schwarz we have that $\mathrm{d}(\tilde{x}, \tilde{\boldsymbol{y}}) \geq \frac{ae\tilde{x} - ae\tilde{\boldsymbol{y}}}{\|ae\|} = \frac{e\tilde{x}}{\sqrt{n}} - \frac{ae\tilde{\boldsymbol{y}}}{a\sqrt{n}}$.

By definition of $\tilde{x}$ we have $e\tilde{x} = \frac{1}{\max\{\alpha,1\}}(1-\epsilon)^2 n$. From the fact the cut $aex \leq b$ is valid for $\boldsymbol{P}$ and $\tilde{\boldsymbol{y}} \in \boldsymbol{P}$, we have $ae\tilde{\boldsymbol{y}} \leq b$. Simple calculations show that $\frac{b}{a\sqrt{n}} \leq \frac{n}{2}(1+\epsilon')$. Plugging these values in we get that $\mathrm{d}(\boldsymbol{P}, \boldsymbol{P}^k) = \mathrm{d}(\tilde{x}, \tilde{\boldsymbol{y}}) \geq \frac{\sqrt{n}}{2}\left(\frac{2(1-\epsilon)^2}{\max\{\alpha,1\}} - (1+\epsilon')\right)$. Theorem 3 follows from the definition of α, ϵ and ϵ'.

1.6 Sparse Cutting-Planes and Extended Formulations

In this section we analyze the relationship between sparse cuts and extended formulations, proving Lemmas 4 and 5.

1.6.1 Proof of Lemma 4

For any set $S \subseteq \mathbb{R}^{n'}$ and $I \subseteq [n']$, define $\tau_I(S) = S + \mathbb{R}^{\bar{I}}$ (recall that $\mathbb{R}^{\bar{I}} = \{x \in \mathbb{R}^{n'} : x_i = 0 \text{ for } i \in I\}$.

Consider $P \subseteq \mathbb{R}^n$ and $Q \subseteq \mathbb{R}^n \times \mathbb{R}^m$ such that $P = \mathrm{proj}_x(Q)$. Given a subset $I \subseteq [n+m]$ we use I_x to denote the indices of I in $[n]$ (i.e. $I_x = I \cap [n]$). We start with the following technical lemma.

Lemma 18. *For every $I \subseteq [n+m]$ we have $\tau_{I_x}(\mathrm{proj}_x(Q)) = \mathrm{proj}_x(\tau_I(Q))$.*

Proof. ($\subseteq$) Take $u_x \in \tau_{I_x}(\mathrm{proj}_x(Q)$; this means that there is $v \in Q$ such that $u_x = \mathrm{proj}_x(v) + d_x$ for some vector $d_x \in \mathbb{R}^n$ with support in I_x. Define $d = (d_x, 0) \in \mathbb{R}^n \times \mathbb{R}^m$, with support in $I_x \subseteq I$. Then $v + d$ belongs to $\tau_I(Q)$ and

$$u_x = \text{proj}_x(v) + d_x = \text{proj}_x(v + d) \in \text{proj}_x(\tau_I(Q)),$$

concluding this part of the proof.

($\supseteq$) Take $u_x \in \text{proj}_x(\tau_I(Q))$. Let $u \in \tau_I(Q)$ be such that $\text{proj}_x(u) = u_x$. By definition, there is $d \in \mathbb{R}^n \times \mathbb{R}^m$ with support in I such that $u + d$ belongs to Q. Then $\text{proj}_x(u+d) = u_x + \text{proj}_x(d)$ belongs to $\text{proj}_x(Q)$; since $\text{proj}_x(d)$ is supported in I_x, we have that u_x belongs to $\tau_{I_x}(\text{proj}_x(Q))$, thus concluding the proof of the lemma. $\square$

The proof of Lemma 4 then follows directly from the above lemma:

$$\begin{aligned}(\text{proj}_x(Q))^k = \bigcap_{J \subseteq [n]} \tau_J(\text{proj}_x(Q)) = \bigcap_{I \subseteq [n+m]} \tau_{I_x}(\text{proj}_x(Q)) \\ \overset{\text{Lemma 18}}{=} \bigcap_{I \subseteq [n+m]} \text{proj}_x(\tau_I(Q)) \supseteq \text{proj}_x\left(\bigcap_{I \subseteq [n+m]} \tau_I(Q)\right) = \text{proj}_x(Q^k).\end{aligned}$$

1.6.2 Proof of Lemma 5

We construct the polytope $Q \subseteq \mathbb{R}^n \times \mathbb{R}^{2n-1}$ as follows. Let T be the complete ordered binary tree of height $\ell + 1$. We let r denote the root node of T. We use $\text{int}(T)$ to denote the set of internal nodes of T, and for an internal node $v \in \text{int}(T)$ we use $\text{left}(v)$ to denote its left child and $\text{right}(v)$ to denote its right child. Let $i(.)$ be a bijection between the leaf nodes of T and the elements of $[n]$. We then define the set Q as the solutions (x, y) to the following:

$$\begin{aligned} & y_r \leq 1 \\ & y_v = y_{\text{left}(v)} + y_{\text{right}(v)}, \ \forall v \in \text{int}(T) \\ & y_v = \frac{2}{n} x_{i(v)}, \ \forall v \in T \setminus \text{int}(T) \\ & y_v \geq 0, \ \forall v \in T \\ & x_i \in [0, 1], \ \forall i \in [n]. \end{aligned} \tag{2}$$

Define $P = \{x \in [0,1]^n : \sum_{i \in [n]} x_i \leq n/2\}$.

Claim 6. *Q is an extended formulation of P, namely* $\mathrm{proj}_x(Q) = P$.

Proof. ($\subseteq$) Take $(\bar{x}, \bar{y}) \in Q$. Let T_j denote the set of nodes of T at level j. It is easy to see (for instance, by reverse induction on j) that $\sum_{v \in T_j} \bar{y}_v = \frac{2}{n} \sum_{i \in [n]} \bar{x}_i$ for all j. In particular, $\bar{y}_r = \frac{2}{n} \sum_{i \in [n]} \bar{x}_i$. Since $\bar{y}_r \leq 1$, we have that $\bar{x} \in P$.

($\supseteq$) Take $\bar{x} \in P$. Define $\bar{y}$ inductively by setting $\bar{y}_{i(v)} = \frac{2}{n} \bar{x}_{i(v)}$ for all leaves v and $\bar{y}_v = \bar{y}_{\mathrm{left}(v)} + \bar{y}_{\mathrm{right}(v)}$ for all internal nodes v. As in the previous paragraph, it is easy to see that $\bar{y}_r = \sum_{i \in [n]} \bar{x}_i \leq n/2$. Therefore, $(\bar{x}, \bar{y})$ belongs to Q. □

Claim 7. *$d(P, P^k) = \sqrt{n/2}$ for all $k \leq n/2$.*

Proof. For every subset $I \subseteq [n]$ of size $n/2$, the incidence vector of I belongs P this implies that, when $k \leq n/2$, the all ones vector e belongs to P^k. It is easy to see that the closest vector in P to e is the vector $\frac{1}{2}e$; since the distance between e and $\frac{1}{2}e$ is $\sqrt{n/2}$, the claim follows. □

Claim 8. *$Q^3 = Q$.*

Proof. Follows directly from the fact that all the equations and inequalities defining Q in (2) have support of size at most 3. □

The proof of Lemma 5 follows directly from the three claims above.

Chapter II

ANALYSIS OF SPARSE CUTTING-PLANES FOR SPARSE MILPS WITH APPLICATIONS TO STOCHASTIC MILPS

2.1 Introduction

2.1.1 Motivation and goal

Cutting-plane technology has become one of the main pillars in the edifice that is a modern state-of-the-art mixed integer linear programming (MILP) solver. Enormous theoretical advances have been made in designing many new families of cutting-planes for general MILPs (see for example, the review papers - [31, 36]). The use of some of these cutting-planes has brought significant speedups in state-of-the-art MILP solvers [7, 29].

While significant progress has been made in developing various families of cutting-planes, lesser understanding has been obtained on the question of cutting-plane selection from a theoretical perspective. Empirically, sparsity of cutting-planes is considered an important determinant in cutting-plane selection. In a recent paper [16], we presented a geometric analysis of quality of sparse cutting-planes as a function of the number of vertices of the integer hull, the dimension of the polytope and the level of sparsity.

In this chapter, we continue to pursue the question of understanding the strength of sparse cutting-planes using completely different techniques, so that we are also able to incorporate the information that most real-life integer programming formulations have sparse constraint matrices. Moreover, the worst-case analysis we present in this chapter depends on parameters that can be determined more easily than the number of vertices of the integer hull as in [16].

In the following paragraphs, we discuss the main aspects of the research direction we consider in this chapter, namely: (i) The fact that solvers prefer using sparse cutting-planes, (ii) the assumption that real-life integer programs have sparse constraint matrix and (iii) why the strength of sparse cutting-planes may depend on the sparsity of the constraint matrix of the IP formulation.

What is the reason for state-of-the-art solvers to bias the selection of cutting-planes towards sparser cuts? Solving a MILP involves solving many linear programs (LP) – one at each node of the tree, and the number of nodes can easily be exponential in dimension. Because linear programming solvers can use various linear algebra routines that are able to take advantage of sparse matrices, adding dense cuts could significantly slow down the solver. In a very revealing study [37], the authors conducted the following experiment: They added a very dense valid equality constraint to other constraints in the LP relaxation at each node while solving IP instances from MIPLIB using CPLEX. This does not change the underlying polyhedron at each node, but makes the constraints dense. They observed approximately 25% increase in time to solve the instances if just 9 constraints were made artificially dense!

Is it reasonable to say that real-life integer programs have sparse constraint matrix? While this is definitely debatable (and surely "counter examples" to this statement can be provided), consider the following statistic: the average number of non-zero entries in the constraint matrix of the instances in the MIPLIB 2010 library is 1.63% and the median is 0.17% (this is excluding the non-negativity or upper bound constraints). Indeed, in our limited experience, we have never seen formulations of MILPs where the matrix is very dense, for example all the variables appearing in all the constraints. Therefore, it would be fair to say that a large number of real-life MILPs will be captured by an analysis that considers only sparse constraint matrices. We formalize later in the chapter how sparsity is measured for our purposes.

Finally, why should we expect that the strength of sparse cutting-planes to be related to the sparsity of the constraint matrix of the MILP formulation? To build some intuition, consider the feasible region of the following MILP:

$$\begin{array}{lll} A^1x^1 & & \leq \; b^1 \\ & A^2x^2 & \leq \; b^2 \\ x^1 \in \mathbb{Z}^{p_1} \times \mathbb{R}^{q_1}, & x^2 \in \mathbb{Z}^{p_2} \times \mathbb{R}^{q_2} & \end{array}$$

Since the constraints are completely disjoint in the x^1 and x^2 variables, the convex hull is obtained by adding valid inequalities in the support of the first p_1+q_1 variables and another set of valid inequalities for the second p_2+q_2 variables. Therefore, sparse cutting-planes, in the sense that their support is not on all the variables, is sufficient to obtain the convex hull. Now one would like to extend such a observation even if the constraints are not entirely decomposable, but "loosely decomposable". Indeed this is the hypothesis that is mentioned in the classical computational paper [13]. This paper solves fairly large scale 0-1 integer programs (up to a few thousand variables) within an hour in the early 1980s, using various preprocessing techniques and the lifted knapsack cover cutting-planes within a cut-and-branch scheme. To quote from this paper:

> "All problems are characterized by sparse constraint matrix with rational data."
>
> "We note that the support of an inequality obtained by lifting (2.7) or (2.9) is contained in the support of the inequality (2.5) ... Therefore, the inequalities that we generate preserve the sparsity of the constraint matrix."

Since the constraints matrices are sparse, most of the cuts that are used in this chapter are sparse. Indeed, one way to view the results we obtain in this chapter is to attempt a mathematical explanation for the empirical observations of quality of sparse cutting-planes obtained in [13]. Finally, we mention here in passing that the quality of Gomory mixed integer cuts were found empirically to be related to the sparsity of LP relaxation optimal tableaux in the paper [17]; however we do not explore particular families of sparse cutting-planes in this chapter.

2.1.2 The nature of results obtained in this chapter

We examine three kinds of MILPs: packing MILPs, covering MILPs, and a more general form of MILPs where the feasible region is arbitrary together with assumptions guaranteeing that the objective function value is non-negative. For each of these problems we do the following:

1. We first present a method to describe the sparsity structure of the constraint matrix.

2. Then we present a method to describe a hierarchy of cutting-planes from very sparse to completely dense. The method for describing the sparsity of the constraint matrix and that for the cuts added are closely related.

3. For a given MILP instance, we assume that once the sparsity structure of the cutting-planes (i.e. the support of the cutting-planes are decided), the strongest (or equivalently all) valid inequalities on these supports are added to the linear programming relaxation and the resulting LP is solved. Call the optimal objective function value of this LP as z^{cut}.

4. All our results are of the following kind: We present bounds on the ratio of z^{cut} and the optimal objective function value of the IP (call this z^I), where the bound depends only on the sparsity structure of the constraint matrix and the support of sparse cuts.

For example, in the packing case, since objective function is of the maximization type, we present an upper bound on $\frac{z^{cut}}{z^I}$ which, we emphasize again, depends entirely on the location of zeros in the constraint matrix and the cuts added and is independent of the actual data of the instance. We note here that the method to describe the sparsity of the matrix and cutting-planes are different for the different types of problems.

We are also able to present examples in the case of all the three types of problems, that show that the bounds we obtain are tight.

Though out this chapter we will constantly refer back to the deterministic equivalent of a two-stage stochastic problem with finitely many realizations of uncertain parameters in the second stage. Such MILPs have naturally sparse formulations. Moreover, sparse cutting-planes, the so-called scenario-specific cuts (or the path inequalities), for such MILPs have been well studied. (See details in Section 2.2). Therefore, any result we obtain for quality of sparse cutting-planes for sparse IPs is applicable is this setting, and this connection allows us to shed some light on the performance of scenario-specific cuts for stochastic MILPs.

We also conduct computational experiments for all these classes of MILPs to study the effectiveness of sparse cutting-planes. Our main observation is the sparse cuts usually perform much better than the worst-case bounds we obtain theoretically.

Outline the chapter: We present all the definitions (of how sparsity is measured, etc.) and the main theoretical results in Section 2.2. Then in Section 2.3 we present results from a empirical study of the same questions. We make concluding remarks in Section 2.4. Section 2.5 provides proofs of all the results presented in Section 2.2.

2.2 Main results

2.2.1 Notation and basic definitions

Given a feasible region of a mixed integer linear program, say P, we denote the convex hull of P by P^I and denote the feasible region of the linear programming relaxation by P^{LP}.

For any natural number n, we denote the set $\{1, \ldots, n\}$ by $[n]$. Given a set V, 2^V

is used to represent its power set.

Definition 19 (Sparse cut on N)**.** *Given the feasible region of a mixed integer linear program (P) with n variables, and a subset of indices $N \subseteq [n]$, we call $\alpha^T x \leq \beta$ a* sparse cut on N *if it is a valid inequality for P^I and the support of α is restricted to variables with index in N, that is $\{i \in [n] \,|\, \alpha_i \neq 0\} \subseteq N$.*

Clarification of the above definition: If $\alpha^T x \leq \beta$ is a sparse cut on N, then $\alpha_i = 0$ for all $i \in [n] \setminus N$, while α_i may also be equal to 0 for some $i \in N$.

Since we are interested in knowing how good of an approximation of P^I is obtained by the addition of all sparse cutting-planes to the linear programming relaxation, we will study the set defined next.

Definition 20 (Sparse closure on N)**.** *Given a feasible region of a mixed integer linear program (P) with n variables and $N \subseteq [n]$, we define the sparse closure on N, denoted as $P^{(N)}$, and defined as*

$$P^{(N)} := P^{LP} \cap \bigcap_{\{(\alpha,\beta)\,|\,\alpha x \leq \beta \text{ is a sparse cut on N }\}} \{x \,|\, \alpha x \leq \beta\}\,.$$

2.2.2 Packing problems

In this section, we present our results on the quality of sparse cutting-planes for packing-type problems, that is problems of the following form:

$$\begin{aligned} \text{(P)} \quad \max \quad & c^T x \\ s.t. \quad & Ax \leq b \\ & x_j \in \mathbb{Z}_+, \forall j \in \mathcal{L} \\ & x_j \in \mathbb{R}_+, \forall j \in [n] \backslash \mathcal{L}, \end{aligned}$$

with $A \in \mathbb{Q}_+^{m \times n}$, $b \in \mathbb{Q}_+^m$, $c \in \mathbb{Q}_+^n$ and $\mathcal{L} \subseteq [n]$.

In order to analyze the quality of sparse cutting-planes for packing problems we will *partition the variables into blocks*. One way to think about this partition is that it allows us to understand the global effect of interactions between blocks of "similar variables". For example, in MIPLIB instances, one can possibly rearrange the rows and columns [8, 6, 38, 1] so that one sees patterns of blocks of variables in the constraint matrices. See Figure 2(a) for an illustration of "observing patterns" in a sparse matrix. Moreover note that in what follows one can always define the blocks to be singletons, that is each block is just a single variable.

The next example illustrates an important class of problems where such partitioning of variables is natural.

Example 21 (Two-stage stochastic problem)**.** *The deterministic equivalent of a two-stage stochastic problem with finitely many realizations of uncertain parameters in the second stage has the following form:*

$$\begin{aligned} \max \quad & c^T y + \sum_{i=1}^{k} (d^i)^T z^i \\ s.t. \quad & Ay \leq b \\ & A^i y + B^i z^i \leq b^i \qquad \forall i \in [k], \end{aligned}$$

where y are the first stage variables and the z^i variables corresponding to each realization in the second stage. Notice there are two types of constraints:

1. *Constraints involving only the first stage variables.*
2. *Constraints involving the first stage variables and second stage variables corresponding to one particular realization of uncertain parameters.*

Note that there are no constraints in the formulation that involve variables corresponding to two different realizations of uncertain parameters.

It is natural to put all the first stage variables y into one block and each of the second stage variables z^i corresponding to one realization of uncertain parameters into a separate block of variables.

To formalize the effect of the interactions between blocks of variables we define a graph that we call as the packing interaction graph. This graph will play an instrumental role in analyzing the strength of sparse cutting-planes.

Definition 22 (Packing interaction graph of A)**.** *Consider a matrix* $A \in \mathbb{Q}^{m\times n}$*. Let* $\mathcal{J} := \{J_1, J_2 \ldots, J_q\}$ *be a partition of the index set of columns of* A *(that is* $[n]$*). We define the* packing interaction graph $G^{pack}_{A,\mathcal{J}} = (V, E)$ *as follows:*

1. *There is one node* $v_j \in V$ *for every part* $J_j \in \mathcal{J}$*.*

2. *For all* $v_i, v_j \in V$*, there is an edge* $(v_i, v_j) \in E$ *if and only if there is a row in* A *with non-zero entries in both parts* J_i *and* J_j*, namely there are* $k \in [m]$*,* $u \in J_i$ *and* $w \in J_j$ *such that* $A_{ku} \neq 0$ *and* $A_{kw} \neq 0$*.*

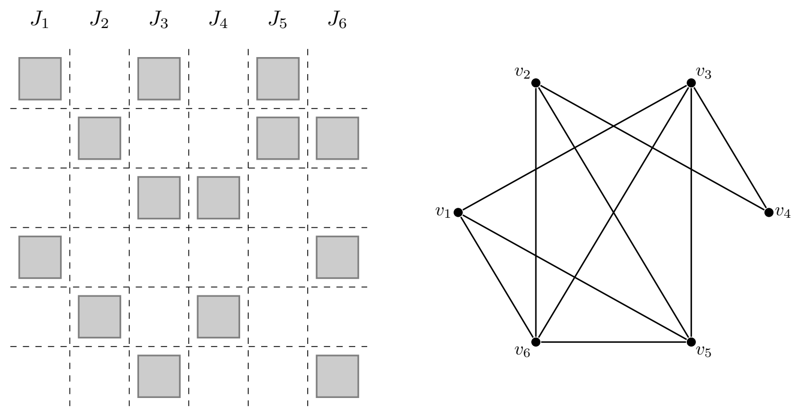
Notice that $G^{\text{pack}}_{A,\mathcal{J}}$ captures the sparsity pattern of the matrix A up to partition $\mathcal{J}$ of columns of the matrix, i.e., this graph ignores the sparsity (or the lack of it) within each of the blocks of columns, but captures the sparsity (or the lack of it) between the blocks of the column. Finally note that if each of the blocks in $\mathcal{J}$ were singletons, then the resulting graph is the intersection graph [20].

Figure 2 illustrate the process of constructing $G^{\text{pack}}_{A,\mathcal{J}}$. Figure 2(a) shows a matrix A, where the columns are partitioned into six variable blocks, the unshaded boxes correspond to zeros in A and the shaded boxes correspond to entries in A that are non-zero. Figure 2(b) shows $G^{\text{pack}}_{A,\mathcal{J}}$.

Example 23 (Two-stage stochastic problem: $G^{\text{pack}}_{A,\mathcal{J}}$)**.** *Given a two-stage stochastic problem with* k *second stage realizations, we partition the variables in* $k+1$ *blocks (as discussed in Example 21). So we have a graph* $G^{pack}_{A,\mathcal{J}}$ *with vertex set* $\{v_1, v_2, \ldots, v_{k+1}\}$ *and edges* $(v_1, v_2), (v_1, v_3), \ldots, (v_1, v_{k+1})$*.*

The sparse cuts we examine will be with respect to the blocks of variables. In other words, while the sparse cuts may be dense with respect to the variables in some

Figure 2: Constructing $G^{\text{pack}}_{A,\mathcal{J}}$.



(a) The matrix A with column partitions: Shaded boxes have non-zero entries.

(b) The resulting graph.

blocks, it can be sparse globally if its support is on very few blocks of variables. To capture this, we use a *support list* to encode which combinations of blocks cuts are allowed to be supported on; we state this in terms of subsets of nodes of the graph $G^{\text{pack}}_{A,\mathcal{J}}$.

Definition 24 (Column block-sparse closure)**.** *Given the problem* (P)*, let* $\mathcal{J} := \{J_1, J_2, \ldots, J_q\}$ *be a partition of the index set of columns of* A *(that is* $[n]$*) and consider the packing interaction graph* $G^{pack}_{A,\mathcal{J}} = (V, E)$*.*

1. *With slight overload in notation, for a set of nodes* $S \subseteq V$ *we say that inequality* $\alpha x \leq \beta$ *is a* sparse cut on S *if it is a sparse cut on its corresponding variables, namely* $\bigcup_{v_j \in S} J_j$*. The closure of these cuts is denoted by* $P^{(S)} := P^{(\bigcup_{v_j \in S} J_j)}$*.*

2. *Given a collection* $\mathcal{V}$ *of subsets of the vertices* V *(the* support list*), we use* $P^{\mathcal{V},P}$ *to denote the closure obtained by adding all sparse cuts on the sets in* $\mathcal{V}$*'s, namely*

$$P^{\mathcal{V},P} := \bigcap_{S \in \mathcal{V}} P^{(S)}.$$

This definition of column block-sparse closure allows us to define various levels of sparsity of cutting-planes that can be analyzed. In particular if $\mathcal{V}$ includes V, then we are considering completely dense cuts and indeed in that case $P^{\mathcal{V},P} = P^I$.

Let $z^I = \max\{c^T x \,|\, x \in P^I\}$ be the IP optimal value and $z^{\mathcal{V},P} = \max\{c^T x \,|\, x \in P^{\mathcal{V},P}\}$ be the optimal value obtained by employing sparse cuts on the support list $\mathcal{V}$. Since we are working with a maximization problem $z^{\mathcal{V},P} \geq z^I$ and our goal is to provide bounds on how much bigger $z^{\mathcal{V},P}$ can be compared to z^I. Moving forward, we will be particularly interested in two types of sparse cut support lists $\mathcal{V}$:

1. *Super sparse closure* ($P^{S.S.} := P^{\mathcal{V},P}$ and $z^{S.S.} := z^{\mathcal{V},P}$): We will consider the sparse cut support list $\mathcal{V} = \{\{v_1\}, \{v_2\}, \{v_3\}, \ldots, \{v_{|V|}\}\}$. We call this the super sparse closure, since once the partition $\mathcal{J}$ is decided, these are the sparsest cuts to be considered.

2. *Natural sparse closure*: Let $A_1, ..., A_m$ be the rows of A. Let V^i be the set of nodes corresponding to block variables that have non-zero entries in A_i (that is $V^i = \{v_u \in V \,|\, A_{ik} \neq 0 \text{ for some } k \in J_u\}$). For the resulting sparse cut support list $\mathcal{V} = \{V^1, V^2, \ldots, V^m\}$, we call the column block-sparse closure as the 'natural' sparse closure (and $P^{N.S.} := P^{\mathcal{V},P}$ and $z^{N.S.} := z^{\mathcal{V},P}$). The reason to consider this case is that once the partition $\mathcal{J}$ is decided, the cuts defining $P^{N.S.}$ most closely resembles the sparsity pattern of the original constraint matrix. To see this, consider the case when $\mathcal{J} = \{\{1\}, \{2\}, \ldots, \{n\}\}$, that is every block is a single variable. In this case, the sparse cut support list $\mathcal{V}$ represents exactly the different sparsity pattern of the various rows of the IP formulation. Indeed the cuts added in [13] satisfied this sparsity pattern.

Example 25 (Two-stage stochastic problem: Specific-scenario cuts)**.** *Consider again the two-stage stochastic problem with k second stage realizations as discussed in Example 23. Consider the cuts on the support of first stages variables together with the variables corresponding to one second stage realization, the so-called* specific-scenario cuts. *Such cutting-planes are well-studied, see for example [22, 39]. Notice that based on the partition $\mathcal{J}$ previously discussed, the closure of all the specific-scenario cuts is precisely equivalent to the natural sparse closure $P^{N.S.}$.*

A standard technique in stochastic integer programming is to make multiple copies of the first stage variables, which are connected through equality constraints, and relax these ("nonanticipativity") equality constraints (via Lagrangian relaxation methods) to produce computationally strong bound [11]. It is straightforward to see that in the case where there is complete recourse, the closure of the specific-scenario cuts or equivalently the natural sparse closure, will give the same bound as this nonanticipativity dual.

To the best of our knowledge there are no known global bounds known on the quality of nonanticipativity dual. The results in this chapter will be able to provide some such bounds.

In order to present our results, we require the following generalizations of standard graph-theoretic notions such a stable sets and chromatic number.

Definition 26 (Mixed stable set subordinate to $\mathcal{V}$)**.** *Let $G = (V, E)$ be a simple graph. Let $\mathcal{V}$ be a collection of subsets of the vertices V. We call a collection of subsets of vertices $\mathcal{M} \subseteq 2^V$ a* mixed stable set subordinate to $\mathcal{V}$ *if the following hold:*

1. *Every set in $\mathcal{M}$ is contained in a set in $\mathcal{V}$*

2. *The sets in $\mathcal{M}$ are pairwise disjoint*

3. *There are no edges of G with endpoints in distinct sets in $\mathcal{M}$.*

Definition 27 (Mixed chromatic number with respect to $\mathcal{V}$)**.** *Consider a simple graph $G = (V, E)$ and a collection $\mathcal{V}$ of subset of vertices.*

- *The* mixed chromatic number $\bar{\eta}^{\mathcal{V}}(G)$ *of G with respect to $\mathcal{V}$ is the smallest* *number of mixed stables sets $\mathcal{M}^1, \ldots, \mathcal{M}^k$ subordinate to $\mathcal{V}$ that cover all vertices of the graph (that is, every vertex $v \in V$ belongs to a set in one of the $\mathcal{M}^i$'s).*

- *(Fractional mixed chromatic number.) Given a mixed stable set $\mathcal{M}$ subordinate to $\mathcal{V}$, let $\chi_{\mathcal{M}} \in \{0,1\}^{|V|}$ denote its incidence vector (that is, for each vertex $v \in V$, $\chi_{\mathcal{M}}(v) = 1$ if v belongs to a set in $\mathcal{M}$, and $\chi_{\mathcal{M}}(v) = 0$ otherwise.) Then we define the* fractional mixed chromatic number

$$\begin{aligned} \eta^{\mathcal{V}}(G) = \ & \min \sum_{\mathcal{M}} y_{\mathcal{M}} \\ & \text{s.t.} \ \sum_{\mathcal{M}} y_{\mathcal{M}} \chi_{\mathcal{M}} \geq \mathbb{1} \\ & \qquad y_{\mathcal{M}} \geq 0 \quad \forall \mathcal{M}, \end{aligned} \tag{3}$$

where the summations range over all mixed stable sets subordinate to $\mathcal{V}$ and $\mathbb{1}$ is the vector in $\mathbb{R}^{|V|}$ of all ones.

Note that when $\mathcal{V}$ corresponds to the super sparse closure $P^{S.S.}$, that is the elements of $\mathcal{V}$ is the collection of singletons, the mixed stable sets subordinate to $\mathcal{V}$ are the usual stable sets in the graph and the (resp. fractional) mixed chromatic number are the usual (resp. fractional) chromatic number.

The following simple example helps to clarify and motivate the definition of mixed stable sets: they identify sets of variables that can be set **independently** and still yield feasible solutions.

Example 28. *Consider the simple packing two-stage stochastic problem:*

$$\begin{aligned} \max \ & c_1 x_1 + c_2 x_2 + c_3 x_3 \\ s.t. \ & a_{11} x_1 + a_{12} x_2 \qquad\qquad \leq b_1 \\ & a_{21} x_1 + \qquad\qquad a_{23} x_3 \leq b_2 \\ & x \in \mathbb{Z}^3_+. \end{aligned}$$

Consider the partition $\mathcal{J} = \{\{1\},\{2\},\{3\}\}$ so that the graph $G^{pack}_{A,\mathcal{J}}$ equals the path $v_2 - v_1 - v_3$. Consider the support list $\mathcal{V} = \{\{1,2\},\{1,3\}\}$ for the "natural sparse closure" setting. Then the maximal mixed stable sets of subordinate to $\mathcal{V}$ are $\mathcal{M}_1 = \{\{1,2\}\}$, $\mathcal{M}_2 = \{\{1,3\}\}$ and $\mathcal{M}_3 = \{\{2\},\{3\}\}$; $\mathcal{M}_4 = \{\{1\}\}$ is a non-maximal mixed stable set.

To see that mixed stable sets identify sets of variables that can be set independently and still yield a feasible solution, for $i = 1,2,3$ let $x^{(i)}$ be the optimal solution to the above packing problem conditioned on $x_j = 0$ for all $j \neq i$; for example $x^{(2)} = (0, \lfloor b_1/a_{12} \rfloor, 0)$ and $x^{(3)} = (0, 0, \lfloor b_2/a_{23} \rfloor)$. Taking the mixed stable set $\mathcal{M}_3 = \{\{2\},\{3\}\}$ we see that the combination of $x^{(2)} + x^{(3)} = (0, \lfloor b_1/a_{12} \rfloor, \lfloor b_2/a_{23} \rfloor)$ is also feasible for the problem.

Moreover, these solutions allow us to upper bound the ratio $z^{\mathcal{V},P}/z^I$, namely the quality of the column block-sparse closure. First, the integer optimum z^I is at least $\max\{c^T(x^{(2)} + x^{(3)}), c^T x^{(1)}\}$. Also, one can show that $z^{\mathcal{V},P} \leq c^T(x^{(2)} + x^{(3)}) + c^T x^{(1)}$ (this uses the fact that actually $x^{(2)} + x^{(3)}$ is the optimal solution for the problem conditioned on $x_1 = 0$, and $x^{(1)}$ the optimal solution conditioned on $x_2 = x_3 = 0$). Together this gives $z^{\mathcal{V},P}/z^I \leq 2$. Notice that the upper bound on $z^{\mathcal{V}}$ is obtained by adding up the solutions corresponding to the sets $\mathcal{M}_3$ and $\mathcal{M}_4$, which together cover all the variables of the problem. Looking at the fractional chromatic number $\eta^{\mathcal{V}}(G^{pack}_{A,\mathcal{J}})$ allow us to provide essentially the best such bound.

Our first result gives a worst-case upper bound on $\frac{z^{\mathcal{V},P}}{z^I}$ that is, surprisingly, independent of the data A, b, c, and depends only on the packing interaction graph $G^{\text{pack}}_{A,\mathcal{J}}$ and the choice of sparse cut support list $\mathcal{V}$.

Theorem 29. *Consider a packing integer program as defined in* (P). *Let $\mathcal{J} \subseteq 2^{[n]}$ be a partition of the index set of columns of A and let $G = G^{pack}_{A,\mathcal{J}} = (V, E)$ be the packing* *interaction graph of A. Then for any sparse cut support list $\mathcal{V} \subseteq 2^V$ we have*

$$z^{\mathcal{V},P} \leq \eta^{\mathcal{V}}(G) \cdot z^I.$$

As discussed before, if we are considering the super sparse closure $P^{S.S.}$, $\eta^{\mathcal{V}}(G)$ is the usual fractional chromatic number. Therefore we obtain the following possibly weaker bound using Brook's theorem [10].

Corollary 30. *Consider a packing integer program as defined in* (P). *Let $\mathcal{J} \subseteq 2^{[n]}$ be a partition of the index set of columns of A and let $G^{pack}_{A,\mathcal{J}}$ be the packing interaction graph of A. Let Δ denote the maximum degree of G. Then we have the following bounds on the optimum value of the super sparse closure $P^{S.S}$:*

1. *If $G^{pack}_{A,\mathcal{J}}$ is not a complete graph or an odd cycle, then*

$$z^{S.S.} \leq \Delta \cdot z^I.$$

2. *If $G^{pack}_{A,\mathcal{J}}$ is a complete graph or an odd cycle, then*

$$z^{S.S.} \leq (\Delta + 1) \cdot z^I.$$

Thus assuming the original IP is sparse and the maximum degree of $G^{\text{pack}}_{A,\mathcal{J}}$ is not very high, the above result says that we get significantly tight bounds using only super sparse cuts. In fact it is easy to show the above Corollary's bounds can be tight when $G^{\text{pack}}_{A,\mathcal{J}}$ is a 3-cycle or a star. We record this result here.

Theorem 31. *For any $\epsilon > 0$:*

1. *There exists a packing integer program as defined in* (P) *and a partition $\mathcal{J} \subseteq 2^{[n]}$ of the index set of columns of A such that the graph $G^{pack}_{A,\mathcal{J}}$ is a 3-cycle and*

$$z^{S.S.} \geq (3 - \epsilon) z^I.$$

2. *(Strength of super sparse cuts for packing two-stage problems) There exists a packing integer program as defined in* (P) *and a partition* $\mathcal{J} \subseteq 2^{[n]}$ *of the index set of columns of* A *such that the graph* $G^{pack}_{A,\mathcal{J}}$ *is a star and*

$$z^{S.S.} \geq (2 - \epsilon) z^I.$$

We mention here in passing that there are many well-known upper bounds on the fractional chromatic number with respect to other graph properties, which also highlight that for sparse graph we expect the fractional chromatic number to be small. For example, let G be a connected graph of max degree Δ and clique number $\omega(G)$. Then

1. $\eta(G) \leq \frac{\omega(G)+\Delta+1}{2}$. ([33])

2. $\eta(G) \geq \Delta$ if and only if G is a complete graph, odd cycle, a graph with $\omega(G) = \Delta$, a square of the 8-cycle, or the strong product of 5-cycle and K_2. Moreover if $\Delta \geq 4$ and G is not any of the graphs listed above, then $\eta(G) \leq \Delta - \frac{2}{67}$. ([26])

One question is whether we can get better bounds using the potentially denser natural sparse cuts. Equivalently, is the fractional chromatic number $\eta^{\mathcal{V}}(G^{\text{pack}}_{A,\mathcal{J}})$ much smaller when we consider the sparse cut support list $\mathcal{V}$ corresponding to the natural sparse closure? We prove results for some special, but important, structures.

Theorem 32 (Natural sparse closure of trees)**.** *Consider a packing integer program as defined in* (P)*. Let* $\mathcal{J} \subseteq 2^{[n]}$ *be a partition of the index set of columns of* A *and let* $G^{pack}_{A,\mathcal{J}}$ *be the packing interaction graph of* A*. Suppose* $G^{pack}_{A,\mathcal{J}}$ *a tree and let* Δ *be its maximum degree. Then:*

$$z^{N.S.} \leq \left(\frac{2\Delta - 1}{\Delta}\right) z^I.$$

Compare this result for natural sparse cuts with the result for super sparse cuts on trees. While with super sparse cuts we are able to get a multiplicative bound of

2 (this is the fractional chromatic number for bipartite graphs), using natural sparse cuts the bound is always strictly less than 2.

Interestingly, this upper bound is tight even when the induced graph $G^{\text{pack}}_{A,\mathcal{J}}$ is a star, which corresponds exactly to the case of stochastic packing programs. The construction of the tight instances are based on special set systems called *affine designs*, where we exploit their particular partition and intersection properties.

Theorem 33 (Tightness of natural sparse closure of trees)**.** *For any $\epsilon > 0$, there exists a packing integer program (P) and a suitable partition $\mathcal{J}$ of variables where $G^{pack}_{A,\mathcal{J}}$ is a star with max degree Δ such that*

$$z^{N.S.} \geq \left(\frac{2\Delta - 1}{\Delta} - \epsilon \right) z^I.$$

As discussed in Example 23, for the case of two-stage stochastic problem with the right choice of $\mathcal{J}$ the packing interaction graph is a star. So we obtain the following corollary of Theorem 32.

Corollary 34 (Specific-scenario cuts for packing two-stage stochastic problems)**.**

Consider a packing-type two-stage stochastic problem with k realization. Then

$$z^{N.S.} \leq \left(\frac{2k - 1}{k} \right) z^I,$$

where $z^{N.S.}$ is the objective function obtained after adding all specific-scenario cuts. Moreover this bound is tight.

We note that the analysis of approximation algorithm for two stage matching problem in the papers [19, 28] is related to the above result. We would explore this relation in next chapter.

Finally we consider the case of natural sparse cutting-planes when $G^{\text{pack}}_{A,\mathcal{J}}$ is a cycle. Interestingly, the fractional mixed chromatic number $\eta^{\mathcal{V}}(G^{\text{pack}}_{A,\mathcal{J}})$ depends on the length of the cycle modulo 3.

Theorem 35 (Natural sparse closure of cycles). *Consider a packing integer program as defined in* (P). *Let* $\mathcal{J} \subseteq 2^{[n]}$ *be a partition of the index set of columns of* A *and let* $G^{pack}_{A,\mathcal{J}}$ *be the packing interaction graph of* A. *If* $G^{pack}_{A,\mathcal{J}}$ *is a cycle of length* K, *then:*

1. *If* $K = 3k, k \in \mathbb{Z}_{++}$, *then* $z^{N.S.} \leq \frac{3}{2} z^I$.
2. *If* $K = 3k+1, k \in \mathbb{Z}_{++}$, *then* $z^{N.S.} \leq \frac{3k+1}{2k} z^I$.
3. *If* $K = 3k+2, k \in \mathbb{Z}_{++}$, *then* $z^{N.S.} \leq \frac{3k+2}{2k+1} z^I$.

Moreover, for any $\epsilon > 0$, *there exists a packing integer program with a suitable partition* $\mathcal{V}$ *of variables, where* $G^{pack}_{A,\mathcal{J}}$ *is a cycle of length* K *such that*

1. *If* $K = 3k, k \in \mathbb{Z}_{++}$, *then* $z^{N.S.} \geq \left(\frac{3}{2} - \epsilon\right) z^I$.
2. *If* $K = 3k+1, k \in \mathbb{Z}_{++}$, *then* $z^{N.S.} \geq \left(\frac{3k+1}{2k} - \epsilon\right) z^I$.
3. *If* $K = 3k+2, k \in \mathbb{Z}_{++}$, *then* $z^{N.S.} \geq \left(\frac{3k+2}{2k+1} - \epsilon\right) z^I$.

All proofs of the above results are presented in Section 2.5.1.

2.2.3 Covering problems

In this section, we present our results on the quality of sparse cutting-planes for covering-type problems, that is problems of the following form:

$$
\begin{aligned}
\text{(C)} \quad \min \quad & c^T x \\
s.t. \quad & Ax \geq b \\
& x_j \in \mathbb{Z}_+, \forall j \in \mathcal{L} \\
& x_j \in \mathbb{R}_+, \forall j \in [n] \backslash \mathcal{L}
\end{aligned}
$$

with $A \in \mathbb{Q}_+^{m \times n}$, $b \in \mathbb{Q}_+^m$, $c \in \mathbb{Q}_+^n$ and $\mathcal{L} \subseteq [n]$. In this case, we would like to prove lower bounds on the objective functions after adding the sparse cutting-planes.

Our first observation is a negative result: super sparse cuts as defined for the packing-type problems can be arbitrarily bad for the case of covering problems. In order to present this result, let formalize the notion of super sparse cuts in this setting. In particular, given an instance of type (C), we assume we partition the variable indices n into $\mathcal{J} = \{J_1, J_2, \ldots, J_q\}$. For all $j \in [q]$ we add all possible cuts that have support on variables with index in J_j. Let $z^{S.S.}$ be the optimal objective function of the resulting LP with cuts.

Theorem 36. *For any constant $M > 0$, there exists a covering integer program (C) and partition $\mathcal{J} := \{J_1, J_2\}$ of $[n]$, such that the corresponding $z^{S.S.}$ and z^I satisfies:*

$$z^I > M \cdot z^{S.S.}.$$

Note that super sparse cuts may have support that are strict subsets of the support on the constraints of the formulation. Theorem 36 suggests that such cutting-planes in the worst case will not produce good bounds for covering problems.

It turns out that in order to analyze sparse cutting-planes for covering problems, the interesting case is when their support is at least the support of the constraints of the original formulation. Moreover, we need to work with a graph that is a "dual" of $G^{\text{pack}}_{A,\mathcal{J}}$, namely it acts on the *rows* of the problem instead of columns. For the matrix A, let A_i be the i^{th} row. We let $\text{supp}(A_i) \subseteq [n]$ be the set of variables which appear in the i^{th} constraint, that is $\text{supp}(A_i) := \{j \in [n] \,|\, A_{ij} \neq 0\}$.

Definition 37 (Covering interaction graph of A)**.** *Consider the matrix $A \in \mathbb{Q}^{m\times n}$. Let $\mathcal{I} = \{I_1, I_2, \ldots, I_p\}$ be a partition of index set of **rows** of A (that is $[m]$). We define the* covering interaction graph $G^{cover}_{A,\mathcal{I}} = (V, E)$ *as follows:*

1. *There is a node $v_i \in V$ for every part $I_i \in \mathcal{I}$.*

2. *For all $v_i, v_j \in V$, there is an edge $(v_i, v_j) \in E$ if and only if there is a **column** of A with non-zero entries in both parts I_i and I_j, namely $\bigcup_{r\in I_i} supp(A_r)$ intersects $\bigcup_{r\in I_j} supp(A_r)$.*

Definition 38 (Row block-sparse closure). *Given the problem* (C)*, let* $\mathcal{I} = \{I_1, I_2, \ldots, I_p\}$ *be a partition of index set of rows of* A *(that is* $[m]$*) and consider the covering interaction graph* $G^{cover}_{A,\mathcal{I}} = (V, E)$*.*

1. *With slight overload in notation, for a set of nodes* $S \subseteq V$ *we say that inequality* $\alpha x \leq \beta$ *is a* sparse cut *on* S *if it is a sparse cut on the union of the support of the rows in* S*, namely* $\alpha x \leq \beta$ *is a sparse cut on* $\bigcup_{v_i \in S} \bigcup_{r \in I_i} supp(A_r)$*. The closure of these cuts is denoted by* $P^{(S)} := P^{(\bigcup_{v_i \in S} \bigcup_{r \in I_i} supp(A_r))}$*.*

2. *Given a collection* $\mathcal{V}$ *of subsets of the vertices* V *(the* row support list*), we use* $P^{\mathcal{V},C}$ *to denote the closure obtained by adding all sparse cuts on the sets in* $\mathcal{V}$*'s, namely*

$$P^{\mathcal{V},C} := \bigcap_{S \in \mathcal{V}} P^{(S)}.$$

Moreover, we define the optimum value over the row block-sparse closure

$$z^{\mathcal{V},C} := \min\left\{c^T x \,|\, x \in P^{\mathcal{V},C}\right\}.$$

Example 39 (Two-stage stochastic problem: $G^{\text{cover}}_{A,\mathcal{I}}$, weak specific-scenario cuts). *Given a two-stage covering stochastic problem with* k *second stage realizations, we partition the rows into* k *blocks (each block consists of constraints between first stage variables only or first stage variables and variables corresponding to one particular realization). So we have a graph* $G^{cover}_{A,\mathcal{I}}$ *with* $V = \{v_1, v_2, \ldots, v_k\}$ *which is a clique. Moreover, if we consider the closure corresponding to the row support list* $\mathcal{V} = \{\{v_1\}, \{v_2\}, \{v_3\}, \ldots, \{v_k\}\}$*, the cuts are quite similar to specific-scenario cuts (although potentially weaker, since the supports of inequalities could possibly be strictly smaller than those allowed in the "specific-scenario cuts" in Section 2.2.2). Therefore we call this closure, the weak specific-scenario closure.*

We now present the main result of this section. In particular, we present a worst-case upper bound on $\frac{z^I}{z^{\mathcal{V},C}}$ that is independent of the data A, b, c, and depends only on $G^{\text{cover}}_{A,\mathcal{I}}$ and the choice of the row support list $\mathcal{V}$. We remind the reader that given a graph G and collection $\mathcal{V}$ of its vertices, $\bar{\eta}^{\mathcal{V}}(G)$ is the mixed chromatic number with respect to $\mathcal{V}$ (see Definition 80).

Theorem 40. *Consider a covering integer programming as defined in* (C). *Let $\mathcal{I} \subseteq 2^{[m]}$ be a partition of the index set of rows of A and let $G = G^{cover}_{A,\mathcal{I}} = (V, E)$ be the covering interaction graph of A. Then for any sparse cut support list $\mathcal{V} \subseteq 2^V$ we have*

$$z^{\mathcal{V},C} \geq \frac{1}{\bar{\eta}^{\mathcal{V}}(G)} \cdot z^I.$$

We make a few comments regarding Theorem 36:

1. While the result of Theorem 40 for covering-type IPs is very "similar" to the result of Theorem 29 for packing-type of IPs, the key ideas in the proofs are different.

2. Like the previous discussion in Section 2.2.2, the chromatic numbers is small for graphs with small max degree. In fact, using Brook's Theorem [10], we can obtain a result very similar to Corollary 30 for the covering case as well.

3. The result of Theorem 36 holds even if upper bounds are present on some or all of the variables (in this case, we also need to assume that the instance is feasible).

Consider the case of two-stage covering stochastic problem with K scenario and $\mathcal{I}$ as defined in Example 39. Since $G^{\text{cover}}_{A,\mathcal{I}}$ is a clique, its chromatic number is K. Therefore we obtain the following corollary of Theorem 40.

Corollary 41 (Weak specific-scenario cuts for covering stochastic problems). *Consider a two-stage covering stochastic problem for K scenario. Let z^* be the optimal objective value obtained after adding all weak specific-scenario cuts. Then*

$$z^* \geq \frac{1}{K} z^I.$$

Next we prove that the bound presented in Corollary 41 is tight (and therefore the result of Theorem 36 is tight for $G^{\text{cover}}_{A,\mathcal{I}}$ being a clique).

Theorem 42. *Let z^* be the optimal objective value obtained after adding all weak specific-scenario cuts for a two-stage covering stochastic problem. Given any $\epsilon > 0$ with $\epsilon < K$, there exists an instance of the covering-type two-stage stochastic problem with K scenarios such that*

$$z^* \leq \frac{1}{(K-\epsilon)} \cdot z^I.$$

The proof of Theorem 42 is perhaps the most involved in this chapter, as the family of instances constructed to prove the above theorem are significantly complicated.

All proofs of the above results are presented in Section 2.5.2.

2.2.4 "Packing-type" problem with arbitrary A matrix

Up until now we have considered packing and covering problems. We now present results under much milder assumptions. In particular, we consider problem (P) with arbitrary matrix $A \in \mathbb{Q}^{m\times n}$ instead of a non-negative matrix (and b is also not assumed to be non-negative). The assumptions we therefore make in this section are: c is a non-negative vector, the variables are non-negative and the objective is of the maximization-type as in (P).

We use the same definition of sparse-cutting planes as for the packing instances considered in Section 2.2.2. All other notation used is also the same as in Section 2.2.2.

As it turns out, even in this significantly more general case, it is possible to obtain tight data-independent bounds on the quality of sparse-cutting-planes. In order to present this result we introduce the notion of *corrected* average constraint density. The reason to introduce this notion is the following: the strength of cuts in this case is determined by the average density, as long as the cuts cover all the variables. Based on this, the corrected average density captures the best bound one can obtain using a given support list.

Definition 43 (Corrected average density)**.** *Let* $\mathcal{V} = \{V^1, V^2, \ldots, V^t\}$ *be the sparse cut support list. For any subset* $\tilde{\mathcal{V}} = \{V^{u_1}, V^{u_2}, ..., V^{u_k}\} \subseteq \mathcal{V}$ *define its* density *as*

$$D(\tilde{\mathcal{V}}) = \frac{1}{k}\sum_{i=1}^{k} |V^{u_i}|.$$

We define the corrected average density of $\mathcal{V}$ *(denoted as* $D_{\mathcal{V}}$*) as maximum value of* $D(\tilde{\mathcal{V}})$ *over all* $\tilde{\mathcal{V}}$*'s that cover* V*, that is,* $\bigcup_{V' \in \tilde{\mathcal{V}}} V' = V$*.*

Note that $D_{\mathcal{V}} \geq 1$ for any choice of sparse cut support list $\mathcal{V}$, and for the trivial list $\mathcal{V} = \{V(G^{\text{pack}}_{A,\mathcal{J}})\}$ that allows fully dense cuts we have $D_{\mathcal{V}} = |V(G^{\text{pack}}_{A,\mathcal{J}})|$. The following is the main result of this section.

Theorem 44. *Let* (P) *be defined by an arbitrary* $A \in \mathbb{Q}^{m\times n}$*,* $b \in \mathbb{Q}^m$*,* $c \in \mathbb{Q}^n_+$*. Let* $\mathcal{J}$ *be a partition of the index set of columns of* A *(that is* $[n]$*). Let* $G^{pack}_{A,\mathcal{J}} = (V, E)$ *be the packing interaction graph of* A *and let* $\mathcal{V}$ *be the sparse cut support list. If the instance is feasible, then:*

$$z^{\mathcal{V},P} \leq (|V| + 1 - D_{\mathcal{V}}) \cdot z^I.$$

Let us see some consequences of Theorem 44. Since $D_{\mathcal{V}} \geq 1$ we obtain the following result.

Corollary 45. *Given* (P)*, with arbitrary* $A \in \mathbb{Q}^{m\times n}$*,* $b \in \mathbb{Q}^m$*,* $c \in \mathbb{Q}^n_+$*. Let* $\mathcal{J}$ *be a partition of the index set of columns of* A *(that is* $[n]$*). If the instance is feasible, then:*

$$z^{\mathcal{V},P} \leq |V| \cdot z^I.$$

It turns out that the bound in Corollary 45 is tight when $G^P_{A,\mathcal{J}}(V, E)$ is a star.

Theorem 46 (Super sparse cuts for two-stage packing-type problem with arbitrary A). *For every $\epsilon > 0$, there exists $A \in \mathbb{Q}^{m\times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n_+$ such that $G^{pack}_{A,\mathcal{J}}$ is a star and*

$$z^{S.S.} \geq |V| \cdot z^I - \epsilon.$$

Now let us consider the case where the sparse cut support list $\mathcal{V}$ corresponds to the natural sparse closure, when $G^P_{A,\mathcal{J}}(V, E)$ is a star or a cycle. Clearly in both these cases we have $D_{\mathcal{V}} = 2$. Therefore we obtain the following Corollary.

Corollary 47 (Natural sparse cuts two-stage packing-type problem with arbitrary A). *Given* (P), *with arbitrary $A \in \mathbb{Q}^{m\times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n_+$. Let $\mathcal{J}$ be a partition of the index set of columns of A (that is $[n]$). Let $G^{pack}_{A,\mathcal{J}} = (V, E)$ be the packing interaction graph of A which is a star or a cycle. If the instance is feasible, then:*

$$z^{N.S.} \leq (|V| - 1) \cdot z^I.$$

We next show that the result of Corollary 47 is tight for $G^{\text{pack}}_{A,\mathcal{J}}$ being a star, which corresponds to two-stage packing-type problem with arbitrary A.

Theorem 48. *For every $t \in \mathbb{Z}_{++}$, there exists an instance of* (P), *with arbitrary $A \in \mathbb{Q}^{m\times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n_+$, a partition of index set of columns $\mathcal{J}$ such that $G^{pack}_{A,\mathcal{J}}$ is a star with $K + 1$ nodes and*

$$z^{N.S.} = ((K + 1) - 1) \cdot z^I.$$

All proofs of the above results are presented in Section 2.5.3.

2.3 Computational experiments

In this section, we present our computational results on the strength of natural sparse closure of pure binary IP.

In Appendix B, we present the algorithm we implemented to estimate z^{cut}, the optimal objective function value of the natural sparse closure.

We describe the random instances we generate in section 2.3.1 and present the results in section 2.3.2. All the experiments have been carried out using CPLEX12.5.

2.3.1 Instance generation

We generated two kinds of problems: two-stage stochastic programming instances and random-graph based instances. We first discuss how we generated the constraint matrix for both types of instances. Then, we discuss how we generated the right-hand side based on the constraint matrix and the objective function.

2.3.1.1 Constraint matrix generation

To simplify the presentation consider the case of packing instances. Covering instances are generated in the same way.

First we generate the packing-type induced graph on `nv` nodes. In case of the two-stage stochastic programming, the packing-type induced graph is a star with `nv` nodes (i.e. $\mathtt{nv} - 1$ realizations of the second stage). For the random-graph based instances, let `p` (parameter) be the probability that an edge exists between any pair of nodes. As a disconnected induced graph implies that the original problem is decomposable, we accept connected graph only.

Next, given the packing-type induced graph, say $G = (V, E)$ (where $\mathtt{nv} = |V|$), we construct a matrix with that can be partitioned into $|E| \times |V|$ blocks with each block of size $\mathtt{sqr} \times \mathtt{sqr}$ (where `sqr` is a parameter). Thus the constraint matrix has $|E| \times \mathtt{sqr}$ rows and $|V| \times \mathtt{sqr}$ columns. The $(i, j)^{\text{th}}$ block is all zeros if edge i is not incident to node j. Else the $(i, j)^{\text{th}}$ block is a randomly generated dense matrix: We assign each entry the distribution of $unif\{1, \mathtt{M}\}$, where `M` is a parameter. For packing-type with arbitrary matrix, we first generate the entry which follows $unif\{1, \mathtt{M}\}$ and then with probability 0.5, we multiply -1. (Thus, each column block has `sqr` variables and there are `sqr` rows with the same support of vertices).

2.3.1.2 Right-hand side generation

To guarantee that the instances generated are non-trivial, we follow the following steps: Randomly select $\mathtt{p}_x$ (parameter) from the set of $\{0.2, 0.4, 0.6, 0.8\}$. A 0-1 vector $x \in \mathbb{R}^n$ is randomly generated where for all $j \in [n]$, $x_j \sim Bernoulli(\mathtt{p}_x)$. A noise vector $\epsilon \in \mathbb{R}^m_+$ is randomly generated as: for all $i \in [m]$, $\epsilon_i \sim unif\{1, \mathtt{M}_\epsilon\}$ ($\mathtt{M}_\epsilon$ is a parameter). For a covering instance, $b = Ax - \epsilon$. Otherwise $b = Ax + \epsilon$.

2.3.1.3 Objective function generation

Every entry in the objective function follows the distribution of $unif\{1, \mathtt{ObjM}\}$. ($\mathtt{ObjM}$ is a parameter.)

2.3.2 Computational results

2.3.2.1 Results for two-stage stochastic programming

We set the number of second-stage scenarios equals to 10. We set $\mathtt{sqr} = 20$, that is the number of variables for both first-stage and second-stage scenarios equals to 20. Also we set $\mathtt{M} = \mathtt{M}_\epsilon = \mathtt{objM} = 10$. We generated 50 instances for each of the three types of problem.

The result for packing-type problem, covering-type problem, and packing-type problem with arbitrary matrix is shown in Table 1, Table 2, and Table 3 respectively.

Table 1: Two-stage packing stochastic programming

Avg. z^{cut}/z^{IP}	Theoretical bound on z^{cut}/z^{IP}
1.00038	1.9

Table 2: Two-stage covering stochastic programming

Avg. z^{IP}/z^{cut}	Theoretical bound on z^{IP}/z^{cut}
1.009	10

Table 3: Two-stage arbitrary packing stochastic programming

Avg. z^{cut}/z^{IP}	Theoretical bound on z^{cut}/z^{IP}
1	10

2.3.2.2 Results for Random Graph based Instances

We set $\mathtt{nv} = 10$, $\mathtt{p} = 0.2$, $\mathtt{sqr} = 20$, $\mathtt{M} = \mathtt{M}_\epsilon = \mathtt{objM} = 10$. For a given random graph we generated 10 random instances, and therefore we generated 50 instances for each of the three types of problem. The result for packing-type problem, covering-type problem, and packing-type problem with arbitrary matrix is shown in Table 4, Table 5, and Table 6 respectively.

Table 4: Random graph based tests on packing problems

Graph Name	Avg. z^{cut}/z^{IP}	bound of z^{cut}/z^{IP}
Ind 1	1.0009	1.8
Ind 2	1.0028	1.75
Ind 3	1.0053	1.667
Ind 4	1.0006	1.75
Ind 5	1.003	2

Table 5: Random graph based tests on covering problems

Graph Name	Avg. z^{IP}/z^{cut}	bound of z^{IP}/z^{cut}
Ind 1	1.0045	2
Ind 2	1.0046	3
Ind 3	1.0059	3
Ind 4	1.0053	3
Ind 5	1.0052	3

Table 6: Random graph based tests on arbitrary packing problems

Graph Name	z^{cut}/z^{IP}	bound of z^{cut}/z^{IP}
Ind 1	1	9
Ind 2	1	9
Ind 3	1	9
Ind 4	1	9
Ind 5	1	9

2.4 Conclusions

In this chapter, we analyzed the strength of sparse cutting-planes for sparse packing, covering and more general MILP instances. The bounds obtained are completely data independent and in particular depend only on the sparsity structure of the constraint matrix and the support of sparse cuts – in this sense, these results truly provide insight into the strength of sparse cuts for sparse MILPs. We have shown that the theoretical bounds are tight in many cases. Especially for packing, the theoretical bounds are quite strong, showing that if we have the correct sparse cutting-planes, then the bound obtained by using these cuts may be quite good.

The computational results are interesting: we observe that for all the types of problems sparse cutting planes perform significantly better than the theoretical prediction. This is perhaps not surprising since the theoretical bounds are data-free and therefore "worst-case" in nature. Hence, the empirical experiments are another justification for the main message of this chapter: In many cases sparse cuts provide very good bounds for sparse IPs.

2.5 *Proofs*

2.5.1 Proofs for packing problems

For any vector $x \in \mathbb{R}^n$ and $N \subseteq [n]$, we use $x|_N$ to denote the projection of x on the coordinates indexed by N.

We first observe that the column sparse closure $P^{(N)}$ can be viewed essentially as the projection of P^I onto the coordinates indexed by N.

Observation 6. *Consider a mixed-integer linear set with n variables. For any $N \subseteq [n]$, let $P^I|_N$ be the projection of P^I onto the indices in N. Then $x \in P^{(N)}$ if and only if $x \in P^{LP}$ and $x|_N \in P^I|_N$.*

Observation 7. *Consider a mixed-integer set of packing type and let $\mathcal{P} \subseteq \mathbb{R}^n$ be the set of feasible solutions. Then for any set of coordinates $N \subseteq [n]$, $x \in \mathbb{R}^N$ belongs to the projection $\mathcal{P}^I|_N$ iff the extension $\tilde{x} \in \mathbb{R}^n$ belongs to $\mathcal{P}^I$, where $\tilde{x}_i = x_i$ if $i \in N$ and $\tilde{x}_i = 0$ if $i \notin N$.*

2.5.1.1 Proof of Theorem 29

Recall we want to show that $z^{\mathcal{V},P} \leq \eta^{\mathcal{V}}(G^{\text{pack}}_{A,\mathcal{J}}) \cdot z^I$. (See Example 28 for a concrete example of how the proof works.) In this section we use P to denote the mixed-integer set corresponding to the packing problem (P).

There is a natural identification of sets of nodes of $G^{\text{pack}}_{A,\mathcal{J}}$ with sets of indices of variables, namely if $\mathcal{J} = \{J_1, J_2, \ldots, J_q\}$ is the given variable index partition and the nodes of $G^{\text{pack}}_{A,\mathcal{J}}$ are $\{v_1, v_2, \ldots, v_q\}$, then the set of vertices $\{v_i\}_{i \in I}$ corresponds to the indices $\bigcup_{i \in I} J_i \subseteq [n]$. We will make use of this correspondence, and in order to make statements precise we use the function $\phi : 2^{V(G^{\text{pack}}_{A,\mathcal{J}})} \rightarrow 2^{[n]}$ to denote this correspondence; with slight abuse of notation, for a singleton set $\{v\}$ we use $\phi(v)$ instead of $\phi(\{v\})$.

Given a set of vertices $S \subseteq V(G^{\text{pack}}_{A,\mathcal{J}})$, let $x^{(S)}$ denote the optimal solution of the packing problem conditioned on all variables x_i outside S taking value 0, or more precisely, $x^{(S)} \in \text{argmax}\{c^T x \mid x \in P^I, x_i = 0 \;\; \forall i \notin \phi(S)\}$ (we will assume without loss of generality that $x^{(S)}$ is integral); since we are working with a packing problem, this is the roughly same as optimizing over the projection of P^I onto the variables in $\phi(S)$ (but notice $x^{(S)}$ lies in the original space).

We start by showing that, roughly speaking, the closure $P^{(S)}$ captures the original packing maximization problem as long as we ignore the coordinates outside S.

Lemma 49. *For any $x \in P^{(S)}$, we have $(c|_{\phi(S)})^T(x|_{\phi(S)}) \le c^T x^{(S)}$.*

Proof. Proof. Given any $x \in P^{(S)}$, Observation 6 implies that $x|_{\phi(S)} \in \text{proj}_{\phi(S)}(P^I)$. Thus, there exists points $\bar{x}^1, \ldots, \bar{x}^k \in P^I$ and $\lambda_1, \ldots, \lambda_k \in [0,1]$, such that $x|_{\phi(S)} = \sum_{i=1}^k \lambda_i \cdot (\bar{x}^i|_{\phi(S)})$ and $\sum_{i=1}^k \lambda_i = 1$. Therefore,

$$(c|_{\phi(S)})^T(x|_{\phi(S)}) = \sum_{i=1}^k \lambda_i \cdot (c|_{\phi(S)})^T(x^i|_{\phi(S)}).$$

To upper bound the right-hand side, consider a point $x^i|_{\phi(S)}$. Let $\tilde{x} \in \mathbb{R}^n$ (the original space) denote the point obtained from $x^i|_{\phi(S)}$ by putting a 0 in all coordinates outside $\phi(S)$, so $\tilde{x}|_{\phi(S)} = x^i|_{\phi(S)}$ and $\tilde{x}|_{[n]\setminus\phi(S)} = \mathbf{0}$. Because P^I is of packing type, notice that $\tilde{x}$ belongs to P^I. The optimality of $x^{(S)}$ then gives that $(c|_{\phi(S)})^T(x^i|_{\phi(S)}) = c^T\tilde{x} \le c^T x^{(S)}$.

Employing this upper bound on the last displayed equation gives

$$(c|_{\phi(S)})^T(x|_{\phi(S)}) \le \sum_{i=1}^k \lambda_i \cdot c^T x^{(S)} = c^T x^{(S)},$$

thus concluding the proof. $\square$

Now we lower bound the packing problem optimum z^I by solutions constructed via mixed stable sets.

Lemma 50. *Given any mixed stable set $\mathcal{M}$ for $G^{pack}_{A,\mathcal{J}}$, the point $\sum_{M\in\mathcal{M}} x^{(M)}$ belongs to P^I. Thus, $z^I \ge \sum_{M\in\mathcal{M}} c^T x^{(M)}$.*

Proof. Proof. We just prove the first statement. First, notice since each $x^{(M)}$ is integral and non-negative, so is the point $\sum_{M\in\mathcal{M}} x^{(M)}$. So consider an inequality $A_i x \le b_i$ in (P). For any two sets $M_1 \neq M_2 \in \mathcal{M}$, notice that the vector A_i either has all zeros on the indices corresponding to M_1 or on the indices corresponding to M_2, namely either $A_i|_{\phi(M_1)} = \mathbf{0}$ or $A_i|_{\phi(M_2)} = \mathbf{0}$. Applying this to all pairs of sets in $\mathcal{M}$, we get that there is only one set $M^* \in \mathcal{M}$ such that $A_i|_{\phi(M^*)}$ is non-zero, which implies that

$$A_i \sum_{M\in\mathcal{M}} x^{(M)} = \sum_{M\in\mathcal{M}} A_i x^{(M)} = \sum_{M\in\mathcal{M}} (A_i|_{\phi(M)})(x^{(M)}|_{\phi(M)}) = A_i x^{(M^*)} \le b_i,$$

where the last inequality follows from the feasibility of $x^{(M^*)}$. Thus, the point $\sum_{M\in\mathcal{M}} x^{(M)}$ satisfies all inequalities $A_i x \le b_i$ of the system (P), concluding the proof. $\square$

Now, we present the proof of Theorem 29.

Proof. Proof of Theorem 29. Let $V = \{v_1, \ldots, v_q\}$ denote the vertices of $G^{\text{pack}}_{A,\mathcal{J}}$, and let MSS denote the set of all mixed stable sets of $G^{\text{pack}}_{A,\mathcal{J}}$ with respect to $\mathcal{V}$. Let $\{y_M\}_{M\in\mathsf{MSS}}$ be an optimal solution of linear problem (3) corresponding to the definition of mixed fractional chromatic number with respect to $\mathcal{V}$, and define $g = \sum_{\mathcal{M}\in\mathsf{MSS}} \chi_{\mathcal{M}} y_{\mathcal{M}} \in \mathbb{R}^q$. Based on the constraints of (3) we have that $g \ge \mathbb{1}$.

We upper bound the optimum $z^{\mathcal{V},P}$ of the column block-sparse closure. For that let

$$x^* = \operatorname{argmax}\{c^T x \,|\, x \in P^{\mathcal{V},P}\},$$

be the optimal solution. Then breaking up the indices of the variables based on the nodes V and using the non-negativity of c and x^*, we have

$$\begin{aligned}
z^{\mathcal{V},P} = c^T x^* = \sum_{j=1}^{q} (c|_{\phi(v_j)})^T (x^*|_{\phi(v_j)}) &\le \sum_{j=1}^{q} g_j \cdot (c|_{\phi(v_j)})^T (x^*|_{\phi(v_j)}) \\
&= \sum_{j=1}^{q} \left(\sum_{\mathcal{M} \in \mathsf{MSS}} (\chi_{\mathcal{M}})_j \cdot y_{\mathcal{M}} \right) \cdot (c|_{\phi(v_j)})^T (x^*|_{\phi(v_j)}) \\
&= \sum_{\mathcal{M} \in \mathsf{MSS}} y_{\mathcal{M}} \cdot \left(\sum_{j=1}^{q} (\chi_{\mathcal{M}})_j \cdot (c|_{\phi(v_j)})^T (x^*|_{\phi(v_j)}) \right) \\
&= \sum_{\mathcal{M} \in \mathsf{MSS}} y_{\mathcal{M}} \cdot \left(\sum_{\{j \,|\, v_j \in \mathcal{M}\}} (c|_{\phi(v_j)})^T (x^*|_{\phi(v_j)}) \right) \\
&= \sum_{\mathcal{M} \in \mathsf{MSS}} y_{\mathcal{M}} \cdot \left(\sum_{S \in \mathcal{M}} (c|_{\phi(S)})^T (x^*|_{\phi(S)}) \right).
\end{aligned}$$

To further upper bound the right-hand side consider some $\mathcal{M} \in \mathsf{MSS}$, some $S \in \mathcal{M}$ and the term $(c|_{\phi(S)})^T (x^*|_{\phi(S)})$. First we claim that x^* belongs to the column block-sparse closure $P^{(S)}$. To see this, first recall from the definition of mixed stable set that there must be a set V_S in the support list $\mathcal{V}$ containing S. Moreover, since $x^* \in P^{\mathcal{V},P} = \bigcap_{V' \in \mathcal{V}} P^{(V')}$, we have $x^* \in P^{(V_S)}$; finally, the monotonicity of closures implies $P^{(S)} \supseteq P^{(V_S)}$, and hence $x^* \in P^{(S)}$. Thus we can employ Lemma 49 to obtain the upper bound $(c|_{\phi(S)})^T (x^*|_{\phi(S)}) \le c^T x^{(S)}$.

Plugging this bound on last displayed inequality and using Lemma 50 we then get

$$z^{\mathcal{V},P} \le \sum_{\mathcal{M} \in \mathsf{MSS}} y_{\mathcal{M}} \cdot \left(\sum_{S \in \mathcal{M}} c^T x^{(S)} \right) \le \sum_{\mathcal{M} \in \mathsf{MSS}} y_{\mathcal{M}} \cdot z^I = \eta^{\mathcal{V}}(G^{\text{pack}}_{A,\mathcal{J}}) \cdot z^I.$$

This concludes the proof. □

2.5.1.2 Proof of Corollary 30 and Theorem 31

Brooks' Theorem [10] is the following result (recall that a proper coloring of a graph is an assignment of colors to the vertices such that no edge has the same color on both endpoints).

Theorem 51 (Brook's Theorem)**.** *Consider a connected graph G of max degree Δ. Then G can be properly colored by Δ colors, except in two cases either when G is a* *complete graph or an odd cycle, in which case it can be properly colored with* $\Delta + 1$ *colors.*

Since the fractional chromatic number is a lower bound on the chromatic number, we obtain Corollary 30.

We now prove Theorem 31.

Proof. Proof of Theorem 31. Recall that super sparse closure corresponds to the support list

$$\mathcal{V} = \{\{v_1\}, \{v_2\}, \dots, \{v_{|V|}\}\}.$$

The proof of both parts is similar.

Part 1. We want to show an example where $z^{SS} \geq (3 - \epsilon)z^I$ for all $\epsilon > 0$ where $G^{\text{pack}}_{A,\mathcal{J}}$ is a 3-cycle. We will construct an integer program with 3 variables and $\mathcal{J} = \{\{1\}, \{2\}, \{3\}\}$. Given $\epsilon > 0$, consider the following packing integer program:

$$\begin{array}{lllll} \max & x_1 & +x_2 & +x_3 & \\ \text{s.t.} & x_1 & +x_2 & & \leq 2 - \frac{2}{3}\epsilon \\ & x_1 & & +x_3 & \leq 2 - \frac{2}{3}\epsilon \\ & & x_2 & +x_3 & \leq 2 - \frac{2}{3}\epsilon \\ & \multicolumn{4}{l}{x_1 \in \mathbb{Z}_+, \quad x_2 \in \mathbb{Z}_+, \quad x_3 \in \mathbb{Z}_+} \end{array} \tag{4}$$

Clearly $G^{\text{pack}}_{A,\mathcal{J}}$ is a 3-cycle. Note that the only valid inequalities that have support on each of the three blocks defined by $\mathcal{J}$ is $x_i \leq 1$ for $i = 1, 2, 3$. Thus, the point $(1 - \frac{\epsilon}{3}, 1 - \frac{\epsilon}{3}, 1 - \frac{\epsilon}{3})$ belongs to the super sparse closure $P^{S.S.}$, and hence the optimum value satisfies $z^{S.S.} \geq 3 - \epsilon$. On the other hand clearly, $z^{IP} = 1$, concluding the proof.

Part 2. We want to show an example where $z^{SS} \geq (2 - \epsilon)z^I$ for all $\epsilon > 0$ where $G^{\text{pack}}_{A,\mathcal{J}}$ is a star. Take $\Delta \in \mathbb{Z}_+$. We construct a packing integer program with 2Δ variables and $\mathcal{J} = \{\{1, 2, \dots, \Delta\}, \{\Delta + 1\}, \{\Delta + 2\}, \dots, \{2\Delta\}\}$. Given $\epsilon > 0$, consider the following integer program

$$\begin{aligned} \max \quad & \textstyle\sum_{i=1}^{2\Delta} x_i \\ \text{s.t.} \quad & x_i + x_{\Delta+i} \leq 2 - \epsilon \ \forall i \in [\Delta] \\ & x \in \mathbb{Z}_+^{2\Delta}. \end{aligned}$$

Clearly $G^{\text{pack}}_{A,\mathcal{J}}$ is a star with Δ leaves. Letting P be the associated mixed integer set of the above integer program, note that the projection of P^I to the first block of variables $\{1, 2, \dots, \Delta\}$ equals $[0, 1]^\Delta$. Also the only valid inequalities that have support on each of the other Δ blocks $\{\Delta + i\}$ is $0 \leq x_i \leq 1$ for $i \in \{\Delta + 1, \dots, 2\Delta\}$. Thus, the point x with $x_i = 1$ for all $i \in [\Delta]$ and $x_i = 1 - \epsilon$ for all $i \in \{\Delta + 1, \dots, 2\Delta\}$ belongs to the super sparse closure $P^{S.S.}$. Thus the optimum $z^{S.S.}$ is at least $2\Delta - \Delta\epsilon$. On the other hand, clearly $z^I = \Delta$, concluding the proof. $\square$

2.5.1.3 Proof of Theorem 32

We prove the desired upper bound $z^{N.S.} \leq \left(\frac{2\Delta-1}{\Delta}\right) \cdot z^I$. Due to Theorem 29, it suffices to upper bound the fractional chromatic number $\eta^{\mathcal{V}}(G^{\text{pack}}_{A,\mathcal{J}}) \leq \frac{2\Delta-1}{\Delta}$ for $\mathcal{V}$ set according to the natural sparse closure setting. Notice however, that in this setting every edge of $E = E(G^{\text{pack}}_{A,\mathcal{J}})$ belongs to some set in $\mathcal{V}$ and vice-verse, and therefore $\eta^{\mathcal{V}}(G^{\text{pack}}_{A,\mathcal{J}}) = \eta^{E}(G^{\text{pack}}_{A,\mathcal{J}})$. Thus, it suffices to prove $\eta^{E}(G^{\text{pack}}_{A,\mathcal{J}}) \leq \frac{2\Delta-1}{\Delta}$.

The following is the main tool for providing an efficient mixed stable set fractional coloring.

Lemma 52. *Let $T = (V, E)$ be a tree of maximum degree Δ. Then there is a collection of $2\Delta - 1$ sets of edges $E_1, E_2, \dots, E_{2\Delta-1}$ and $2\Delta - 1$ sets of nodes $V_1, V_2, \dots, V_{2\Delta-1}$ satisfying the following:*

1. *For each $i \in [2\Delta-1]$ the collection $E_i \cup V_i$ is a mixed stable set for T subordinate to E*

2. *Each node of T is covered exactly Δ times by the collection of mixed stable sets* $\{E_i \cup V_i\}_{i\in[2\Delta-1]}$.

Proof. Proof. If T consists of a single edge, then $\Delta = 1$ and we can simply set E_1 to be the edge of T and set $V_1 = \emptyset$ to get the desired sets. So assume that T has at least one internal node.

In order to simplify the proof we make all the degrees the same: construct the tree T' from T by adding new leaves to all internal nodes of T so that now every internal node of T' has degree exactly Δ. We will construct the desired sets $\{E'_i\}_{i\in[2\Delta-1]}$ and $\{V'_i\}_{i\in[2\Delta-1]}$ for T' via a coloring argument reminiscent of the proof of Brook's Theorem (although not the same argument).

Pick any internal node v_0 of T' and root this tree at v_0. We label all edges and leaf nodes of T' with numbers in $[2\Delta - 1]$ according to the following BFS procedure (we use the standard meaning of "parent", "child", "depth" (where v_0 has depth 0, L is the maximum depth of any node), etc. for rooted trees):

Label each of the Δ edges incident to the root v_0 with a distinct label
for $i = 1$ to L **do**
  for every vertex v of depth i **do**
    Let S denote the set of labels assigned to all the edges incident to the parent of v and notice that $|S| = \Delta$
    if v is an internal node **then**
      Label the $\Delta - 1$ edges of v to its children with distinct labels from the set $[2\Delta - 1] \setminus S$
    else
      Assign **all** $\Delta - 1$ labels $[2\Delta - 1] \setminus S$ to v.

Then for all $j \in [2\Delta - 1]$, let E'_j (resp. V'_j) be set of edges (resp. nodes) of T' that have label j (notice that vertices have multiple labels).

It follows directly from the labeling procedure that each set $E'_i \cup V'_i$ is a mixed stable set of T' (and clearly subordinate to the edges of T'). Now to see that each node v of T' is covered exactly Δ times by the collection $\{E'_i \cup V'_i\}_{i\in[2\Delta-1]}$ we consider 2 cases: If v is an internal node, then by construction of T' it has degree exactly Δ

and since $\bigcup_i E'_i = E(T')$ it is covered Δ times by the collection $\{E'_i\}_i$ and 0 times by the collection $\{V'_i\}_i$, giving the desired result. On the other hand, if v is a leaf of T', then it is covered once by the set E'_i where i is the label of the only edge incident to v, covered by no other set E'_j, and covered by the $\Delta - 1$ sets V'_j corresponding to the labels of v. Thus, the sets $\{E'_i\}_i$ and $\{V'_i\}_i$ satisfy the desired properties with respect to the modified tree T'.

Now to get the desired sets for the original tree T, we just remove the nodes in $T' \setminus T$ from the sets $E'_i \cup V'_i$: for each $\{v, v'\}$ with $v \in V(T)$ and $v' \notin V(T)$ that belongs to $E'_i \cup V'_i$, replace it with the singleton $\{v\}$; denote the set obtained by $E_i \cup V_i$ (concretely, E_i is the set of pairs in this collection and V_i is the set of singletons in this collection). Notice that this replacement procedure does not add repeated singletons: this is because $E'_i \cup V'_i$ contains only disjoint edges (the labeling scheme above does not assign color i to two intersecting edges) and if it contains an edge (v, v') with $v' \notin V(T)$ then this implies that v is an internal node of T and hence the singleton $\{v\}$ does not belong to $E'_i \cup V'_i$.

It follows directly from this replacement operation that the sets $E_i \cup V_i$'s are mixed stable sets for T subordinate to $E(T)$ and that still each node of T is covered exactly Δ times by them. This concludes the proof. $\square$

The upper bound in Theorem 32 the follows from the following corollary.

Corollary 53. *Let H be a tree of maximum degree Δ. Then $\eta^E(H) \leq \frac{2\Delta-1}{\Delta}$.*

Proof. Proof. Consider the mixed stable sets $\mathcal{M}_i = E_i \cup V_i$ (for $i \in [2\Delta - 1]$) of H obtained from Lemma 52. Since each vertex of H is covered exactly Δ times by $\{\mathcal{M}_i\}_{i \in 2\Delta-1}$, we have that setting $y_{\mathcal{M}_i} = \frac{1}{\Delta}$ for all i (and $y_{\mathcal{M}} = 0$ otherwise) yields a feasible solution for the mixed fractional chromatic number program (3) of value $\frac{2\Delta-1}{\Delta}$, proving the result. $\square$

2.5.1.4 Proof of Theorem 33

Fix $\epsilon > 0$; we construct an instance where $z^{N.S.} \geq \left(\frac{2\Delta-1}{\Delta} - \epsilon\right) \cdot z^I$. The construction require the existence of the so-called *affine designs*.

Definition 54. *Given $n \in \mathbb{Z}_{++}$, we call an* affine n-design *a collection $\mathcal{F}_1, \ldots, \mathcal{F}_n$ where each $\mathcal{F}_i$ is a family of n-subsets of $[n^2]$ satisfying:*

1. *For any $i \in [n]$, the sets in $\mathcal{F}_i$ partition $[n^2]$*
2. *For any $i \neq j \in [n]$ and $A \in \mathcal{F}_i$ and $B \in \mathcal{F}_j$, we have $|A \cap B| \leq 1$.*

Theorem 55 ([12], Part VII, Point 2.17)**.** *For every prime n, an affine n-design exists.*

So consider a prime number $n \geq \Delta$ and let $\mathcal{F}_1, \ldots, \mathcal{F}_n$ be an affine n-design. For a set $A \in \mathcal{F}_i$ we use $\chi_A \in \{0,1\}^{n^2}$ to denote the indicator vector of the set A.

We will construct a packing IP in $\mathbb{R}_+^{n^2+\Delta}$ and partition the $n^2 + \Delta$ variables into $\Delta + 1$ blocks $\mathcal{J} = \{J_0, \ldots, J_\Delta\}$ by setting $J_0 = \{1, \ldots, n^2\}$ and $J_i = \{n^2 + i\}$, for $i = 1, \ldots, \Delta$. To simplify the notation we use $x \in \mathbb{R}^{n^2}$ to represent the variables in J_0, and y_i, $i = 1, \ldots, \Delta$, to represent the variables in J_i respectively.

Let P_i be the polytope in $\mathbb{R}^{n^2+\Delta}$ given by the convex hull of the points

$$\left\{ (x, y_1, \ldots, y_\Delta) \in \{0,1\}^{n^2+\Delta} \;\middle|\; \sum_j x_j \leq n \text{ and, if } y_i = 1, \text{ then } x \leq \chi_A \text{ for some } A \in \mathcal{F}_i \right\};$$

explicitly, this is the set of solutions satisfying

$$\begin{aligned} &\sum_j x_j \leq n \\ &x_a + x_b + y_i \leq 2 \qquad \forall a \in A, b \in B,\ A \neq B \in \mathcal{F}_i \\ &(x, y_1, \ldots, y_\Delta) \in [0,1]^{n^2+\Delta}. \end{aligned} \tag{5}$$

Then the desired IP (P) is obtained by considering the integer solutions common to all these polytopes:

$$\max \sum_{j \in [n^2]} x_j + \left(\frac{n-1}{\Delta - 1}\right) \sum_{j \in [\Delta]} y_j$$
$$(x, y_1, \ldots, y_\Delta) \in \bigcap_{i \in [\Delta]} P_i \cap \mathbb{Z}^{n^2+\Delta}.$$

Again we get the following interpretation for the feasible solutions for this problem: in any solution $(x, y) \in \{0,1\}^{n^2+\Delta}$, $\sum_j x_j \leq n$ and for all $i \in [\Delta]$

$$\text{if } y_i = 1, \text{ then } x \leq \chi_A \text{ for some set } A \in \mathcal{F}_i \tag{6}$$

Let P denote the integer set corresponding to this problem. From the explicit description of the P_i's we see that this is a packing integer program whose induced graph $G^{\text{pack}}_{A,\mathcal{J}}$ is a star with maximum degree Δ.

The intuition behind this construction is the following: first, maximizing the objective function over just $P_i \cap \mathbb{Z}^{n^2+\Delta}$ (or equivalently over P_i) gives value $n + \left(\frac{n-1}{\Delta-1}\right) \cdot \Delta \approx 2n$ (by taking $x = \chi_A$ for any $A \in \mathcal{F}_i$, $y_j = 1$ for all j). Moreover, recall that the natural sparse closure w.r.t. $\mathcal{J}$ of the full program (P) uses cuts that are only supported in (x, y_i), for $i \in [\Delta]$; thus, roughly speaking, this closure sees each $P_i \cap \mathbb{Z}^{n^2+\Delta}$ independently, and not really capturing the fact they are being intersected. Thus, optimizing over the natural sparse closure w.r.t. $\mathcal{J}$ still gives value $\approx 2n$. However, due to the fact the sets across the design's $\mathcal{F}_i$'s are almost disjoint, intersecting the regions $P_i \cap \mathbb{Z}^{n^2+\Delta}$ kills most of the solutions. A bit more precisely, the almost disjointness in the affine design and expression (6) imply that the best solution either sets many of the y_i's to 1 and almost all x_j's to 0, or sets all x_j's to 1 and few y_i's to 0; these solutions gives value $\approx n$. This gives the desired gap of ≈ 2 between the natural sparse closure and the original IP.

To make this formal, we start with the following lemma.

Lemma 56. *Setting $x = (\frac{1}{n}, \dots, \frac{1}{n})$ and $y_j = 1$ for all j gives a feasible solution to the natural sparse closure $P^{N.S}$. Thus, $z^{N.S.} \geq n + \left(\frac{n-1}{\Delta-1}\right) \cdot \Delta$.*

Proof. Proof. Let $\bar{x} = (\frac{1}{n}, \dots, \frac{1}{n})$ and $\bar{y} = (1, \dots, 1)$ denote the desired solution.

We claim that it suffices to prove that $(\bar{x}, e^i)$ belongs to P^I for all i, (where e^i is the ith canonical basis vector in $\mathbb{R}^\Delta$). To see that, first notice that the natural sparse closure w.r.t. $\mathcal{J}$ is $P^{N.S.} = \bigcap_{i \in [\Delta]} P^{(x,y_i)}$, where we use $P^{(x,y_i)}$ to denote the sparse closure of P with cuts on variables (x, y_i) (see Definition 20). Using Observations 6 and 7, it suffices to show $(\bar{x}, \bar{y}) \in P^{LP}$ and $(\bar{x}, e^i) \in P^I$. The former condition can be easily verified via equation (5), so it suffices to show $(\bar{x}, e^i) \in P^I$ for all i.

So fix $i \in [\Delta]$. Consider the collection $\mathcal{F}_i$ and a point of the form (χ_A, e^i) for any set $A \in \mathcal{F}_i$. By definition of P_i, notice that (χ_A, e^i) belongs to $P_i \cap \mathbb{Z}^{n^2+\Delta}$. Moreover, notice that for $j \neq i$ we also have $(\chi_A, e^i) \in P_j$: this follows from the facts $\sum_j (\chi_A)_j \leq n$ and $e^i_j = 0$. Thus, we have $(\chi_A, e^i) \in P^I = \bigcap_{j \in [\Delta]} P_j \cap \mathbb{Z}^{n^2+\Delta}$. Then the average $\sum_{A \in \mathcal{F}_i} \frac{1}{n}(\chi_A, e^i)$ belongs to P^I; since the sets in $\mathcal{F}_i$ form a partition of $[n^2]$, $\sum_{A \in \mathcal{F}_i} \chi_A = (1, \dots, 1)$, and hence the average is $\sum_{A \in \mathcal{F}_i} \frac{1}{n}(\chi_A, e^i) = (\bar{x}, e^i) \in P^I$. This concludes the proof. □

The next step is to understand P better.

Lemma 57. *For any solution $(x, y_1, \dots, y_\Delta) \in P$ with $\sum_{i \in [\Delta]} y_i \geq 2$ we have $\sum_{j \in [n^2]} x_j \leq 1$.*

Proof. Proof. Consider $p \neq q$ such that $y_p = y_q = 1$. By definition of P, we have that the solution $(x, y_1, \dots, y_\Delta)$ belongs to P_p and P_q. Since $y_p = y_q = 1$, this means that there are sets $A \in \mathcal{F}_p$ and $B \in \mathcal{F}_q$ such that $x \leq \chi_A$ and $x \leq \chi_B$, which further implies $x \leq \chi_{A \cap B}$. But by definition of an affine design $|A \cap B| \leq 1$, and hence $\sum_{j \in [\Delta]} x_j \leq 1$. This concludes the proof. □

Corollary 58. *We have $z^I \leq \frac{n\Delta - 1}{\Delta - 1}$.*

Proof. Proof. Consider any feasible solution $(x, y_1, \ldots, y_\Delta)$ to P^I. From the first constraint in (5) we have $\sum_j x_j \leq n$. Thus, if $\sum_i y_i \leq 1$, the solution has value at most $n + \left(\frac{n-1}{\Delta-1}\right) = \frac{n\Delta-1}{\Delta-1}$; on the other hand, using Lemma 57, if $\sum_i y_i \geq 2$ then the solution has value at most $1 + \left(\frac{n-1}{\Delta-1}\right)\Delta = \frac{n\Delta-1}{\Delta-1}$. Together these give the desired upper bound. □

Lemma 56 and Corollary 58 give that

$$\frac{z^{N.S.}}{z^I} \geq \left(n + \left(\frac{n-1}{\Delta-1}\right) \cdot \Delta\right) \frac{\Delta-1}{n\Delta-1} = \frac{2n\Delta - n - \Delta}{n\Delta - 1} = \frac{2\Delta - 1 - \Delta/n}{\Delta - 1/n}.$$

Since $\lim_{n\to\infty} \frac{2\Delta-1-\Delta/n}{\Delta-1/n} = \frac{2\Delta-1}{\Delta}$, for a sufficiently large choice of n we get $z^{N.S.} \geq \left(\frac{2\Delta-1}{\Delta} - \epsilon\right) z^I$. This concludes the proof of Theorem 33.

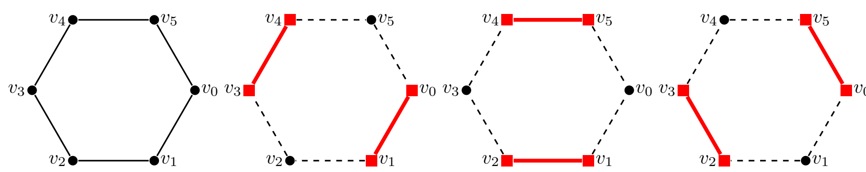
2.5.1.5 Proof of the first part of Theorem 35: upper bound on $z^{N.S.}$

Consider the packing interaction graph $G^{\text{pack}}_{A,\mathcal{J}}$, which is a cycle of length K. Notice that the natural sparse closure in this case corresponds to considering the support list $\mathcal{V}$ being simply the edges of $G^{\text{pack}}_{A,\mathcal{J}}$. Thus, to prove the first part of Theorem 35 is suffices to upper bound the fractional mixed chromatic number $\eta^{E(G^{\text{pack}}_{A,\mathcal{J}})}(G^{\text{pack}}_{A,\mathcal{J}})$.

We can work more abstractly to simplify things: let $H = (V, E)$ be the cycle $v_0 - v_1 - \ldots - v_{K-1} - v_0$ on K nodes, and we need to upper bound $\eta^E(H)$. To further simplify the notation, we identify v_i with $v_{i \pmod K}$ for $i \geq K$. We consider the different cases depending on $K \pmod 3$.

Case 1: $K = 3k$, $k \in \mathbb{Z}_{++}$. For $i = 0, 1, 2$, let $\mathcal{M}_i$ denote the set of edges $\{v_j, v_{j+1}\}$ where $j = i \pmod 3$. It is clear that each $\mathcal{M}_i$ is a mixed stable set for H subordinate to E. Moreover, since $\bigcup_{i=0}^{3} \mathcal{M}_i = E$ covers each node of H exactly twice, we can find a solution for the fractional mixed chromatic number LP (3) by setting $y_{\mathcal{M}_i} = \frac{1}{2}$ for $i = 0, 1, 2$. This gives the desired bound $\eta^E(H) \leq \frac{3}{2}$.

Figure 3: Constructions of all mixed stable sets for 6-cycle



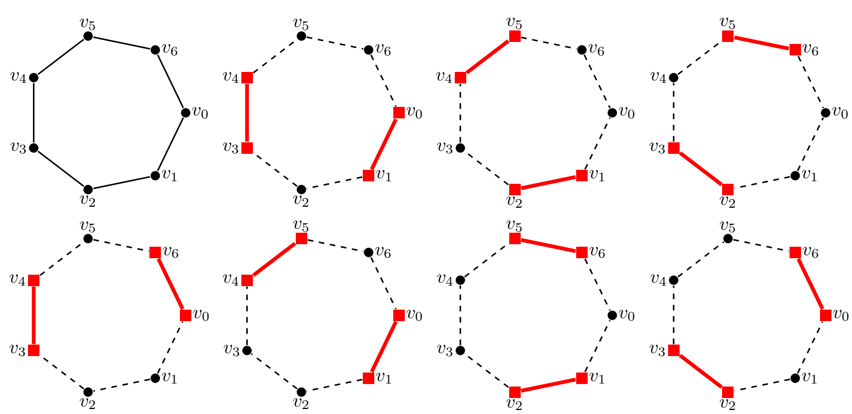
Case 2: $K = 3k + 1$, $k \in \mathbb{Z}_{++}$. We show $\eta^E(H) \leq \frac{3k+1}{2k}$. If $k = 1$, we have that H is a 4-cycle and define $\mathcal{M}_0 = \{(v_0, v_1)\}$, $\mathcal{M}_1 = \{(v_1, v_2)\}$, $\mathcal{M}_2 = \{(v_2, v_3\})$ and $\mathcal{M}_3 = \{(v_3, v_0)\}$. Clearly these $\mathcal{M}_i$'s are mixed stable sets for H subordinate to E and $\bigcup_i \mathcal{M}_i$ covers each node of H exactly twice; then as in the previous case, this gives $\eta^E(H) \leq \frac{4}{2} = 2 = \frac{3k+1}{2k}$.

For $k \geq 2$, define

$$\mathcal{M}_i = \big\{\{v_i, v_{i+1}\}, \{v_{i+3}, v_{i+4}\}, \ldots, \{v_{3(k-2)+i}, v_{3(k-2)+i+1}\}, \{v_{3(k-1)+i}, v_{3(k-1)+i+1}\}\big\}$$

for $i = \{0, \ldots, 3k\}$. It is straightforward to check that each $\mathcal{M}_i$ is a mixed stable set subordinate to E and that $\bigcup_{i=0}^{3k} \mathcal{M}_i$ covers every node exactly $2k$ times. Thus again we get $\eta^E(H) \leq \frac{3k+1}{2k}$.

Figure 4: Constructions of all mixed stable sets for 7-cycle

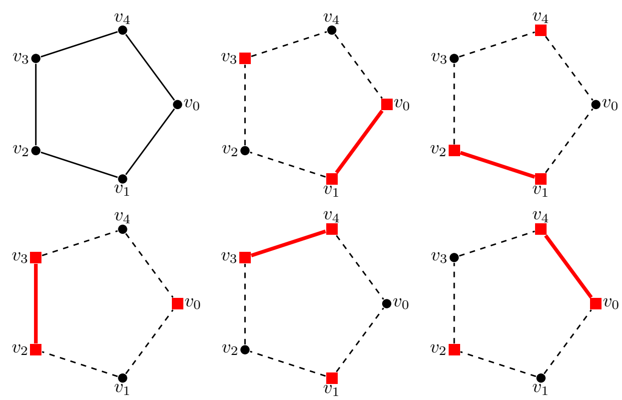


Case 3: $K = 3k+2$, $k \in \mathbb{Z}_{++}$. Let

$$\mathcal{M}_i = \big\{\{v_i, v_{i+1}\}, \{v_{i+3}, v_{i+4}\}, \ldots, \{v_{3(k-2)+i}, v_{3(k-2)+i+1}\}, \{v_{3(k-1)+i}, v_{3(k-1)+i+1}\}, \{v_{3k+i}\}\big\}$$

for $i = \{0, \ldots, 3k+1\}$. It is straightforward to check that each $\mathcal{M}_i$ is a mixed stable set subordinate to E and that $\bigcup_{i=0}^{3k+1} \mathcal{M}_i$ covers every node exactly $2k+1$ times. Thus we have $\eta^E(H) \leq \frac{3k+2}{2k+1}$. This concludes the proof of the first part of the theorem.

Figure 5: Constructions of all mixed stable sets for 5-cycle



2.5.1.6 Proof of second part of Theorem 35: tight instances

The construction of the tight instances is similar to the one used in Theorem 33. So consider a prime number $n \geq K$ and let $\mathcal{F}_1, \ldots, \mathcal{F}_n$ be an affine n-design. For a set $A \in \mathcal{F}_i$ again we use $\chi_A \in \{0,1\}^{n^2}$ to denote the indicator vector of the set A.

We will construct a packing IP with Kn^2 variables, which are partitioned into K equally sized blocks $\mathcal{J} = \{J_0, \ldots, J_{K-1}\}$, namely $J_i = \{n^2 i, n^2 i+1, \ldots, n^2 i + n^2 - 1\}$. To simplify the notation, we use $x^i \in \mathbb{R}^{n^2}$ to represent the variables corresponding to J_i, so a solution of the IP has the form $(x^0, \ldots, x^{K-1})$. For $i \geq K$, we use x^i to denote $x^{i \pmod K}$.

First, define the integer set $Q = \{x \in \{0,1\}^{n^2} \mid \mathbb{1}^T x \leq n\}$. Then, for $i \in$

$\{0, \ldots, K-1\}$ let P_i be the polytope in $\mathbb{R}^{Kn^2}$ given by the convex hull of the points

$$\left\{ (x^0, \ldots, x^{K-1}) \in Q^K \;\middle|\; \begin{array}{l} \text{if } x^i \neq 0, \text{ then } x^{i+1} \leq \chi_A \text{ for some } A \in \mathcal{F}_i, \text{ and} \\ \text{if } x^{i+1} \neq 0, \text{ then } x^i \leq \chi_A \text{ for some } A \in \mathcal{F}_i \end{array} \right\};$$

explicitly, this is the set of solutions satisfying

$$\begin{aligned} & \mathbb{1}^T x^j \leq n && \forall j \\ & x^i_a + x^i_b + x^{i+1}_c \leq 2 && \forall a \in A, b \in B,\ A \neq B \in \mathcal{F}_i,\ \forall c \\ & x^{i+1}_a + x^{i+1}_b + x^i_c \leq 2 && \forall a \in A, b \in B,\ A \neq B \in \mathcal{F}_i,\ \forall c \\ & (x^0, \ldots, x^{K-1}) \in [0,1]^{Kn^2}. \end{aligned} \tag{7}$$

Then the desired IP (P) is obtained by considering the integer solutions common to all these polytopes:

$$\max \sum_{i=0}^{K-1} \mathbb{1}^T x^i$$
$$(x^0, \ldots, x^{K-1}) \in \bigcap_{i=0}^{K-1} P_i \cap \mathbb{Z}^{Kn^2}.$$

Again we get the following interpretation for the feasible solutions for this problem: in any solution $(x^0, \ldots, x^{K-1}) \in \{0,1\}^{Kn^2}$, $\mathbb{1}^T x^i \leq n$ for all i, and also for all i

$$\begin{aligned} & \text{if } x^i \neq 0, \text{ then } x^{i+1} \leq \chi_A \text{ for some set } A \in \mathcal{F}_i, \text{ and} \\ & \text{if } x^{i+1} \neq 0, \text{ then } x^i \leq \chi_A \text{ for some set } A \in \mathcal{F}_i. \end{aligned} \tag{8}$$

Let P denote the integer set corresponding to this problem. From the explicit description of the P_i's we see that this is a packing integer program whose induced graph $G^{\text{pack}}_{A,\mathcal{J}}$ is a K-cycle.

We now consider the natural sparse closure $P^{N.S.}$ and the integer hull P^I for this problem and lower bound the ratio $z^{N.S.}/z^I$. For that, given $x = (x^0, \ldots, x^{K-1})$, let $\mathsf{high}(x) = \{i \mid \mathbb{1}^T x^i \geq 2\}$, namely the set of block of variables with "high" value. We say that three integers are *adjacent mod* K if they are of the form $i \pmod K, i + 1 \pmod K, i + 2 \pmod K$.

Lemma 59. *For any solution $x \in P$, the set* $\mathsf{high}(x)$ *does not contain any three adjacent mod K integers.*

Proof. Proof. By contradiction, assume that $\mathsf{high}(x)$ contains the integers $i \pmod K, i+1 \pmod K$, and $i+2 \pmod K$. In particular, all of x^i, x^{i+1} and x^{i+2} are different from 0, and hence expression (8) implies that $x^{i+1} \leq \chi_A$ and $x^{i+1} \leq \chi_B$ for some $A \in \mathcal{F}_{i \pmod K}$ and $B \in \mathcal{F}_{i+1 \pmod K}$; this implies that $x^{i+1} \leq \chi_{A \cap B}$. But by definition of affine design, we have $|A \cap B| \leq 1$, and hence $\mathbb{1}^T x^{i+1} \leq 1$, reaching a contradiction. □

The following lemma can be easily checked.

Lemma 60. *Let S be a subset of $\{0, \ldots, K-1\}$ that does not contain any three adjacent mod K integers. Then: (i) if $K = 3k$ or $K = 3k+1$ for $k \in \mathbb{Z}_{++}$ we have $|S| \leq 2k$, and; (ii) if $3k+2$ for $k \in \mathbb{Z}_{++}$ we have $|S| \leq 2k+1$.*

Lemma 61. *The optimal value of the integer program* (P) *can be upper bounded as follows: if $K = 3k$ or $K = 3k+1$ for $k \in \mathbb{Z}_{++}$, $z^I \leq (n-1) \cdot 2k + K$; if $K = 3k+2$ for $k \in \mathbb{Z}_{++}$, $z^I \leq (n-1) \cdot (2k+1) + K$.*

Proof. Proof. Let $\bar{x} = (\bar{x}^0, \ldots, \bar{x}^{K-1})$ be an optimal solution to (P). Using the fact that $\mathbb{1}^T \bar{x}^i \leq n$ and the definition of $\mathsf{high}(\bar{x})$ we get

$$
\begin{aligned}
z^I = \sum_{i=0}^{K-1} \mathbb{1}^T \bar{x}^i &= \sum_{i \in \mathsf{high}(\bar{x})} \mathbb{1}^T \bar{x}^i + \sum_{i \notin \mathsf{high}(\bar{x})} \mathbb{1}^T \bar{x}^i \\
&\leq n \cdot |\mathsf{high}(\bar{x})| + K - |\mathsf{high}(\bar{x})| = (n-1) \cdot |\mathsf{high}(\bar{x})| + K.
\end{aligned}
$$

Upper bounding $|\mathsf{high}(\bar{x})|$ using Lemmas 59 and 60 gives the desired result. □

Lemma 62. *The point $\bar{x} = (\frac{1}{n}\mathbb{1}, \ldots, \frac{1}{n}\mathbb{1})$ is a feasible solution to the natural sparse closure $P^{N.S}$. Thus, $z^{N.S.} \geq Kn$.*

Proof. Proof. To simplify the notation, let $\mathsf{zero}^i(x, x') \in \mathbb{R}^{n^2} \times \ldots \mathbb{R}^{n^2}$ denote the vector

$$(0, \ldots, 0, x, x', 0, \ldots, 0)$$

where x is in the ith position and x' is in position $i + 1 \pmod K$.

We claim that it suffices to prove that $\mathsf{zero}^i(\mathbb{1}/n, \mathbb{1}/n)$ belongs to P^I for all i. To see that, first notice that the natural sparse closure w.r.t. $\mathcal{J}$ is $P^{N.S.} = \bigcap_{i\in 0}^{K-1} P^{(x^i,x^{i+1})}$, where we use $P^{(x^i,x^{i+1})}$ to denote the sparse closure of P on variables (x^i, x^{i+1}). Using Observations 6 and 7, it suffices to show $\bar{x} \in P^{LP}$ and $\mathsf{zero}^i(\mathbb{1}/n, \mathbb{1}/n) \in P^I$. The former condition can be easily verified via equation (5), so it suffices to show $\mathsf{zero}^i(\mathbb{1}/n, \mathbb{1}/n) \in P^I$.

So fix i. Consider the collection $\mathcal{F}_i$. By the definition of P_i, for each $A, B \in \mathcal{F}_i$ the point $\mathsf{zero}^i(\chi_A, \chi_B)$ belongs to $P_i \cap \mathbb{Z}^{Kn^2}$. If also follows directly from the definition of P_j that $\mathsf{zero}^i(\chi_A, \chi_B) \in P_j$ for all $j \neq i$. Thus, we have $\mathsf{zero}^i(\chi_A, \chi_B) \in P^I = \bigcap_{j=0}^{K-1} P_j \cap \mathbb{Z}^{n^2+\Delta}$. Then the following average belongs to P^I:

$$\sum_{A\in\mathcal{F}_i} \frac{1}{n} \sum_{B\in\mathcal{F}_i} \frac{1}{n}\mathsf{zero}^i(\chi_A, \chi_B) = \sum_{A\in\mathcal{F}_i} \frac{1}{n}\mathsf{zero}^i\left(\chi_A, \sum_{B\in\mathcal{F}_i} \frac{1}{n}\chi_B\right) = \mathsf{zero}^i\left(\sum_{A\in\mathcal{F}_i} \frac{1}{n}, \sum_{B\in\mathcal{F}_i} \frac{1}{n}\right).$$

Recalling that $\sum_{A\in\mathcal{F}_i} \chi_A = \mathbb{1}$, this average is $\mathsf{zero}^i\left(\frac{1}{n}\mathbb{1}, \frac{1}{n}\mathbb{1}\right) \in P^I$. This concludes the proof. $\square$

Putting Lemmas 61 and 62 together, we get that if $K = 3k$ for $k \in \mathbb{Z}_{++}$, $\frac{z^{N.S.}}{z^I} \geq \frac{Kn}{(n-1)\cdot 2k+K} = \frac{n}{(n-1)\cdot(2/3)+1} = \frac{1}{2/3+1/3n}$. Since $\lim_{n\to\infty} \frac{1}{2/3+1/3n} = \frac{3}{2}$, for sufficiently large n we have $z^{N.S.} \geq z^I(\frac{3}{2} - \epsilon)$, proving this part of the theorem. The other cases of $K \pmod 3$ are similar. This concludes the proof.

2.5.2 Proof for covering problem

2.5.2.1 Proof of Theorem 36

In order to prove Theorem 36, we begin with a classical bad example for the LP relaxation of the set cover problem.

Definition 63 (Special set covering problem (SSC)). *Consider $q \in \mathbb{Z}_+$. The ground set of the set cover problem will be $\{0,1\}^q$, and the covering sets $S(v) = \{u \in \{0,1\}^q \setminus \{0\} \mid v^T u = 1 \ (mod\ 2)\}$ for $v \in \{0,1\}^q$. Then the Special Set Covering (SSC(q)) problem is defined by:*

$$
\begin{aligned}
(SSC(q)) \quad \min & \sum_{v \in \{0,1\}^q} x_v \\
s.t. & \sum_{v : u \in S(v)} x_v \geq 1 \qquad \forall u \in \{0,1\}^q \\
& x_v \in \{0,1\} \qquad\qquad \forall v \in \{0,1\}^q.
\end{aligned}
$$

We refer to the left-hand matrix of SSC as A^q.

Theorem 64 ([30]). *The IP optimal value of $SSC(q)$ is at least q, while the LP relaxation optimal value is at most 2.*

We are now ready to present the proof of Theorem 36. For that, we consider the following problem:

$$
\begin{aligned}
(DSC(q)) \quad \min & \sum_{v \in \{0,1\}^q} x_v + \sum_{v \in \{0,1\}^q} y_v \\
s.t. \ & A^q x + A^q y \geq \mathbb{1} \\
& x, y \in \{0,1\}^{2^q}
\end{aligned}
$$

We first argue that $DSC(q)$ preserves the gap between IP and LP from $SSC(q)$.

Lemma 65. *The IP optimal value of $DSC(q)$ is at least q, while the LP relaxation optimal value is at most 2.*

Proof. Proof. ($z^{LP} \leq 2$): By Theorem 64, there exists a feasible solution $\bar{x}$ of the LP relaxation of $SSC(q)$ such that $\sum_v \bar{x}_v \leq 2$. Then $(\bar{x}, 0)$ is a feasible solution of the LP relaxation of $DSC(q)$, giving the desired bound.

($z^{IP} \geq q$): Assume by contradiction that $(\bar{x}, \bar{y})$ is a feasible solution of $DSC(q)$ with objective function less than q. Note that if $\bar{x}_v = \bar{y}_v = 1$, then we may set $\bar{y}_v = 0$

and still obtain a feasible solution with a better objective function value; similarly, if $\bar{x}_v = 0$ and $\bar{y}_v = 1$ we may set $\bar{x}_v = 1$ and $\bar{y}_v = 0$ and obtain a feasible solution with same objective value. Therefore, we may assume that $\bar{y} = 0$. In this case, $\bar{x}$ is a feasible solution of SSC with objective function less than q, contradicting the statement of Theorem 64. $\square$

We consider a partition on the columns of $DSC(q)$ into two blocks: $\mathcal{J} = \{J_1, J_1\}$ where J_1 corresponding to variables x and J_2 corresponding to variables y. To complete the proof of the theorem it is sufficient to prove that the super sparse closure optimal value $z^{S.S.}$ of $DSC(q)$ is equal to optimal LP value z^{LP} of $DSC(q)$.

Lemma 66. *$z^{S.S} = z^{LP}$ for $DSC(q)$.*

Proof. Proof. Let P be the integer set for $DSC(q)$. Since $P^{S.S} = P^{(J_1)} \cap P^{(J_2)}$, Observation 6 gives that $(\bar{x}, \bar{y}) \in P^{S.S}$ iff $(\bar{x}, \bar{y}) \in P^{LP}$ and $\bar{x} \in P^I|_{J_1}$ and $\bar{y} \in P^I|_{J_2}$. But since P is of covering-type and $SSC(q)$ is feasible, we have that $P^I|_{J_i} = [0,1]^{2^q}$, and thus $(\bar{x}, \bar{y}) \in P^{S.S.}$ iff $(\bar{x}, \bar{y}) \in P^{LP}$. This concludes the proof. $\square$

2.5.2.2 Proof of Theorem 40

Consider a covering problem (C). As in the packing case, there is an identification of sets of nodes of $G^{\text{cover}}_{A,\mathcal{I}}$ with sets of indices of variables (the "indices in the union of their support"), namely if $\mathcal{I} = \{I_1, I_2, \dots, I_q\}$ is the given row index partition and the nodes of $G^{\text{cover}}_{A,\mathcal{I}}$ are $\{v_1, v_2, \dots, v_q\}$, then the set of vertices $\{v_i\}_{i \in I}$ corresponds to the indices $\bigcup_{i \in I} \bigcup_{r \in I_i} \text{supp}(A_r) \subseteq [n]$. We will make use of this correspondence, and in order to make statements precise we use the function usupp : $2^{V(G^{\text{cover}}_{A,\mathcal{I}})} \to 2^{[n]}$ to denote this correspondence; with slight abuse of notation, for a singleton set $\{v\}$ we use usupp(v) instead of usupp$(\{v\})$.

Given a set of vertices $S \subseteq V(G^{\text{cover}}_{A,\mathcal{I}})$, let $x^{(S)}$ be the optimal solution of the covering problem projected to the variables relative to S, namely $x^{(S)} \in \text{argmin}\{(c|_{\text{usupp}(S)})^T y \mid y \in P^I|_{\text{usupp}(S)}\}$. Also, let $\mathsf{zero}^S(x^{(S)}) \in \mathbb{R}^n$ denote the solution appended by zeros in the original space, namely $\mathsf{zero}^S(x^{(S)})_i = x_i^{(S)}$ if $i \in \text{usupp}(S)$ and $\mathsf{zero}^S(x^{(S)})_i = 0$ if $i \notin \text{usupp}(S)$.

Notice the following important property of $\mathsf{zero}^S(x^{(S)})$ (denote $S = \{v_i\}_{i\in I}$): for any row $r \in \bigcup_{i\in I} I_i$, since the support of A_r is contained in usupp(S), the constraint $A_r x \geq b_r$ is valid for $P^I|_{\text{usupp}(S)}$; therefore $A_r \mathsf{zero}^S(x^{(S)}) = A_r|_{\text{usupp}(S)} x^{(S)} \geq b_r$. This gives the following.

Observation 8. *For any subset $S = \{v_i\}_{i\in I}$ of nodes of $G^{cover}_{A,\mathcal{I}}$ and any row $r \in \bigcup_{i\in I} I_i$,*

$$A_r \mathsf{zero}^S(x^{(S)}) \geq b_r.$$

We start by showing that the solutions $x^{(M)}$, for M in a mixed stable set $\mathcal{M}$, can be used to provide a lower bound on the optimal value of $P^{\mathcal{V},C}$.

Lemma 67. *Let $\mathcal{M}$ be a mixed stable set for $G^{cover}_{A,\mathcal{I}}$ subordinate to $\mathcal{V}$. Then*

$$z^{\mathcal{V},C} \geq \sum_{M\in\mathcal{M}} (c|_{usupp(M)})^T x^{(M)}.$$

Proof. Proof. Consider an optimal solution $x^* \in \text{argmin}\{c^T x \,|\, x \in P^{\mathcal{V},C}\}$ of the row block-sparse closure. Since $P^{\mathcal{V},C} = \bigcap_{S\in\mathcal{V}} P^{(S)}$, Observation 6 implies that $x^*|_{\text{usupp}(S)} \in P^I|_{\text{usupp}(S)}$ for all $S \in \mathcal{V}$. Moreover, since for every set M in the mixed stable set $\mathcal{M}$ there is $S \in \mathcal{V}$ containing M, this implies that $x^*|_{\text{usupp}(M)} \in P^I|_{\text{usupp}(M)}$ for all $M \in \mathcal{M}$. Then by the optimality of $x^{(M)}$, we get $(c|_{\text{usupp}(M)})^T(x^*|_{\text{usupp}(M)}) \geq (c|_{\text{usupp}(M)})^T x^{(M)}$ for all $M \in \mathcal{M}$.

Then we can decompose the optimal solution x^* based on the variables usupp(M) and use the non-negativity of c:

$$z^{\mathcal{V},C} = c^T x^* = \sum_{M\in\mathcal{M}} (c|_{\text{usupp}(M)})^T(x^*|_{\text{usupp}(M)}) + \sum_{i\notin \bigcup_{M\in\mathcal{M}} \text{usupp}(M)} c_i x_i^* \geq \sum_{M\in\mathcal{M}} (c|_{\text{usupp}(M)})^T x^{(M)},$$

where the first equality uses the fact that if $M_1, M_2 \in \mathcal{M}$, then usupp$(M_1) \cap$ usupp$(M_2) = \emptyset$. This concludes the proof. $\square$

Now show how to put solutions $x^{(M)}$ together to get a feasible solution for the covering problem, thus providing an upper bound on z^I. Recall the definition of mixed chromatic number $\bar{\eta} = \bar{\eta}^{\mathcal{V}}(G^{\text{cover}}_{A,\mathcal{I}})$ and consider covering mixed stable sets $\mathcal{M}_1, \ldots, \mathcal{M}_{\bar{\eta}}$ (i.e., $V(G^{\text{cover}}_{A,\mathcal{I}}) = \bigcup_i \bigcup_{M \in \mathcal{M}_i} M$).

Define $u \in \mathbb{R}^n$ as the pointwise maximum of the solutions $\{\mathsf{zero}^M(x^{(M)}))\}_{i, M \in \mathcal{M}_i}$. Since the matrix A in the problem is non-negative, Observation 8 implies that u is a feasible solution for the covering problem (C). Thus, using the non-negativity of c and of the $\mathsf{zero}^M(x^{(M)})$'s:

$$z^I \leq c^T u \leq \sum_{i, M \in \mathcal{M}_i} c^T \mathsf{zero}^M(x^{(M)}) = \sum_{i, M \in \mathcal{M}_i} (c|_{\text{usupp}(M)})^T x^{(M)} \leq \sum_i z^{\mathcal{V},C} = \bar{\eta} \cdot z^{\mathcal{V},C},$$

where the first inequality follows from definition of z^I and feasibility of u, the second inequality follows from non-negativity of c, and the last inequality follows from Lemma 67. This concludes the proof of Theorem 40.

2.5.2.3 Proof of Theorem 42

Now we prove Theorem 42 by constructing a covering instance. Since the construction is quite involved, we start with an example.

Example of the construction. We exemplify the construction for $K = 2$ and with a worse gap, and then we generalize/strengthen it (the discussion here will be somewhat informal). In this case the covering IP is the following (notice the indices of the x variables in the different constraints):

$$\min \sum_i x_i + \infty \cdot (y_1 + y_2) \tag{9}$$

$$s.t. \quad \left[\begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} x_1 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} x_2 \right] + \left[\begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} x_3 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} x_4 \right] + \mathbb{1} \cdot y_1 \geq \mathbb{1} \tag{10}$$

$$\left[\begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} x_1 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} x_3 \right] + \left[\begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} x_2 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} x_4 \right] + \mathbb{1} \cdot y_2 \geq \mathbb{1} \tag{11}$$

$$x \in \mathbb{Z}^4_+, \quad y \in \mathbb{Z}^2_+. \tag{12}$$

We will use the partition of rows $\mathcal{I} = \{I_1, I_2\}$, where $I_1 = \{1, 2, 3, 4\}$ (so corresponds to the first sets of covering constraints) and $I_2 = \{5, 6, 7, 8\}$.

The only (minimal) ways to satisfy the first set of constraints is to set either $x_1 = x_2 = 1$ (and all else to 0), or $x_3 = x_4 = 1$ (and all else to 0), or $y_1 = 1$ (and all else to 0); because of the cost of the y variables, actually we will always have $y = 0$ in an optimal solution. To satisfy the second set of constraints the situation is similar, but the indices on the x variables are permuted so that we need $x_1 = x_3 = 1$ or $x_2 = x_4 = 1$. So the best way to satisfy both of the constraints **simultaneously** is to set almost all x variables to 1 (actually we can just set $x_1 = x_2 = x_3 = 1$). This gives cost of 3 for the IP.

Now consider optimizing over the weak specific-scenario cuts closure $P^{\mathcal{V},C}$ (where the row support list is $\mathcal{V} = \{\{v_1\}, \{v_2\}\}$), i.e., the closure corresponding to the cuts on (x, y_1) variables and on (x, y_2) variables. Since the y_i variable can be used to satisfy the ith set of covering constraints, it is easy to see that the only nondominated (x, y_1)-cuts are the ones implied only the first set of covering constraints (10), and similarly the only nondominated (x, y_2)-cuts are the ones implied only by the second set of covering constraints (11). Thus, the point $x_1 = x_2 = x_3 = x_4 = \frac{1}{2}$, $y_1 = y_2 = 0$ belongs to $P^{\mathcal{V},C}$, giving $z^* = z^{\mathcal{V},C} \leq 2$.

Together, these observations give that $\frac{z^I}{z^{\mathcal{V},C}} \geq 3/2$.

General construction. We start with the special set system that is used to define the columns of the covering program.

Lemma 68. *Let $n \in \mathbb{Z}_{++}$. There is a collection $\mathcal{G}^1, \mathcal{G}^2, \ldots, \mathcal{G}^n$ with the following properties:*

1. *For each $i \in [n]$, $\mathcal{G}^i$ is a partition of $[n^n]$ and each set $G \in \mathcal{G}^i$ has size n^{n-1}.*

2. *For any selection $G^1 \in \mathcal{G}^1, G^2 \in \mathcal{G}^2, \ldots, G^n \in \mathcal{G}^n$, the intersection $\bigcap_{i=1}^n G^i$ is non-empty.*

Proof. Proof. Since $|[n^n]| = |[n]^n|$, let $g : [n]^n \to [n^n]$ be any bijection between the two sets. For $j \in [n]$, define the set $G^i_j = \{g(u) \,|\, u \in [n]^n, u_i = j\}$. Define $\mathcal{G}^i = \{G^i_j \,|\, j \in [n]\}$. It is easy to check the following properties:

1. Given i, for any $j \in [n]$, $\left|G^i_j\right| = n^{n-1}$ and $\bigcup_{j=1}^n G^i_j = \{g(u) \,|\, u \in [n]^n\} = [n^n]$.

2. For a selection $G^1_{j_1} \in \mathcal{G}^1, G^2_{j_2} \in \mathcal{G}^2, \ldots, G^n_{j_n} \in \mathcal{G}^n$, consider $u = (j_1, j_2, \ldots, j_n)$. Then according to the definition, $g(u) \in \bigcap_{i=1}^n G^i_{j_i}$, so the intersection of these sets is non-empty.

This concludes the proof. $\square$

Lemma 69. *Let $n \in \mathbb{Z}_{++}$. Consider a collection $\mathcal{G}^1, \ldots, \mathcal{G}^n$ satisfying the properties of Lemma 68, and consider $\bar{\mathcal{G}}^i \subseteq \mathcal{G}^i$ for $i = 1, \ldots, n$. If the sets in $\bigcup_{i=1}^n \bar{\mathcal{G}}^i$ cover the whole of $[n^n]$, then there is $i \in [n]$ such that $\bar{\mathcal{G}}^i = \mathcal{G}^i$.*

Proof. Proof. By contradiction, suppose there is $G^1_{j_1} \in \mathcal{G}^1 \setminus \bar{\mathcal{G}}^1, \ldots, G^n_{j_n} \in \mathcal{G}^n \setminus \bar{\mathcal{G}}^n$. Then part 2 of Lemma 68, there exists an element $u \in \bigcup_{i=1}^n G^i_{j_i}$, and since the sets in $\mathcal{G}^i$ partition $[n^n]$ this means that u is not covered by sets in $\bar{\mathcal{G}}^i$, for all i; then $\bigcup_{i=1}^n \bar{\mathcal{G}}^i$ does not cover $[n^n]$, a contradiction. □

Now pick a prime number $n \geq \max\{K, 2\}$. We will construct an instance with variables $x_1, \ldots, x_{n^2}$ and $y_1, \ldots, y_K$, and each row-block will have n^n constraints. Let $\mathcal{G}^1, \ldots, \mathcal{G}^n$ be a collection satisfying the properties from Lemma 68, and let $\mathcal{F}_1, \ldots, \mathcal{F}_n$ be an affine n-design (we remind the readers – each $\mathcal{F}_i$ partitions $[n^2]$ with n-subsets, and for $A \in \mathcal{F}_i, B \in \mathcal{F}_j$ with $i \neq j$, $|A \cap B| \leq 1$); this affine design will be used to "permute" the indices of the x variables from one set of covering constraints to the next (see example above). We consider the explicit enumeration $\mathcal{F}_i = \{F^1_i, \ldots, F^n_i\}$.

For $k \in [K]$, we define the set

$$P_k = \left\{ (x, y) \in \{0,1\}^{n^2+K} \;\middle|\; \sum_{i=1}^n \sum_{j \in F^k_i} A^k_j x_j + \mathbb{1} \cdot y_k \geq \mathbb{1} \right\},$$

where the set of vectors $\{A^k_j\}_{j \in F^k_i}$ is equal to the set of vectors $\{\chi_{G^i_{j'}}\}_{j' \in [n]}$; we note that it is not important which A^k_j is assigned to which $\chi_{G^i_{j'}}$.

Then the covering integer program we consider is the following:

$$\begin{aligned}
\min \quad & \sum_{j=1}^{n^2} x_j + n^n \cdot \sum_{k=1}^{K} y_k \\
s.t. \quad & \sum_{i=1}^n \sum_{j \in F^k_i} A^k_j x_j + \mathbb{1} \cdot y_k \geq \mathbb{1} \qquad \forall k \in [K] \\
& (x, y) \in \{0,1\}^{n^2+K}.
\end{aligned}$$

(or equivalently $(x, y) \in \bigcap_{k \in [K]} P_k$). Let P denote the set of solutions for this problem.

Notice that this program has K sets of n^n covering constraints. We consider the partition of the covering constraints $\mathcal{I} = \{I_1, \ldots, I_K\}$ where $I_k = \{(k-1)n^n + 1, \ldots, kn^n\}$ (so I_k corresponds to P_k). It is easy to see that the covering interaction graph $G^{\text{cover}}_{A,\mathcal{I}}$ is a clique.

For $\mathcal{V} = \{\{v_1\}, \ldots, \{v_K\}\}$, remember that $z^* = z^{\mathcal{V},C}$. Therefore, we want to show that $\frac{z^I}{z^{\mathcal{V},C}} \geq K - \epsilon$ (for sufficiently large n). We start by analyzing each P_k.

Lemma 70. *A vector $(\bar{x}, 0) \in \{0,1\}^{n^2} \times \{0,1\}^K$ belongs to P_k if and only if there is F_i^k such that $\bar{x}_j = 1$ for all $j \in F_i^k$.*

Proof. Proof. $(\Rightarrow)$ Let $\bar{\mathcal{G}}^i$ be the subset of the sets in $\mathcal{G}^i$ picked by $\bar{x}$, namely G_t^i belongs to $\bar{\mathcal{G}}^i$ iff $\chi_{G^i_{j'}} = A_j^k$ for some j with $\bar{x}_j = 1$. Since $\bar{y} = 0$, the fact that $(\bar{x}, \bar{y})$ belongs to P_k implies that the sets in $\bigcup_{i=1}^n \bar{\mathcal{G}}^i$ must cover the whole of $[n^n]$. Lemma 69 then implies that there is one $\bar{\mathcal{G}}^i$ that equals $\mathcal{G}^i$, which translates to having $\bar{x}_j = 1$ for all $j \in F_i^k$.

$(\Leftarrow)$ This follows from the fact that the sets in $\mathcal{G}^i$ cover the whole of $[n^n]$. $\square$

We can use this to lower bound the optimal value z^I of the covering program P.

Lemma 71. *$z^I \geq Kn - K^2$.*

Proof. Proof. First, we claim that if $(\bar{x}, 0) \in P$, then $\sum_{j \in [n^2]} \bar{x}_j \geq Kn - K^2$. Let $S \subseteq [n^2]$ be the support of $\bar{x}$, so it is equivalent to show $|S| \geq nK - K^2$. Since $(\bar{x}, 0) \in P = \bigcap_{k=1}^K P_k$, using Lemma 70 we have that for every $k \in [K]$ there is $i(k)$ such that S contains $F_{i(k)}^k$, so $S \supseteq \bigcup_{k=1}^K F_{i(k)}^k$. By the inclusion-exclusion principle, we have that $|S| \geq \left|\bigcup_{k=1}^K F_{i(k)}^k\right| \geq \sum_{k=1}^K |F_{i(k)}^k| - \sum_{k \neq k'} |F_{i(k)}^k \cap F_{i(k')}^{k'}|$. Using the definition of an affine n-design, get the lower bound $|S| \geq nK - K(K-1) \geq nK - K^2$.

Now consider any solution $(\bar{x}, \bar{y}) \in P$. If $\bar{y} = 0$, we have just shown that this solution has value at least $Kn - K^2$; if $\bar{y} \neq 0$, this solution has value at least $n^n > Kn - K^2$. This concludes the proof. $\square$

Finally we upper bound the optimal value of the $z^{\mathcal{V},C}$.

Lemma 72. *$z^{\mathcal{V},C} \leq n$.*

Proof. Proof. It suffices to show that the point $(\bar{x}, \bar{y}) = \left(\frac{1}{n}\mathbb{1}, 0\right) \in P^{\mathcal{V},C}$. Recall that $P^{\mathcal{V},C} = \bigcap_{k\in[K]} P^{(\{v_k\})}$, so we show $(\bar{x}, \bar{y})$ belongs to all $P^{(\{v_k\})}$'s. Note that $(\bar{x}, \bar{y})$ satisfies the linear programming relaxation; therefore, using Observation 6, to show that $(\bar{x}, \bar{y})$ belongs to $P^{(\{v_k\})}$ it suffices to prove that $(\bar{x}, \bar{y}_k) \in P^I|_{(x,y_k)}$, where we use $P^I|_{(x,y_k)}$ to denote the projection onto the variables (x, y_k).

Consider the following points (x^u, y), for $u \in [n]$, constructed as:

$$
\begin{aligned}
y_k &= 0, \\
y_{k'} &= 1 \ \forall\ k' \in [K] \setminus \{k\}, \\
x^u_j &= \begin{cases} 1 & \text{if } j \in F^k_u \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}
$$

It is straightforward to verify that $(x^u, y) \in P$ for $u \in [n]$. Thus, the average $\frac{1}{n}\sum_{u\in[n]}(x^u, y)$ belongs to P^I. It then follows that $(\bar{x}, \bar{y}_k) = (\frac{1}{n}\mathbb{1}, 0)|_{(x,y_k)}$ belongs to $P^I|_{(x,y_k)}$. This concludes the proof. $\square$

Putting Lemmas 71 and 72 together we get $\frac{z^I}{z^{S.S.}} \geq \frac{Kn-K^2}{n} = K - \frac{K^2}{n}$. For large enough n, we get $\frac{z^I}{z^{\mathcal{V},C}} \geq K - \epsilon$. This concludes the proof of Theorem 42.

2.5.3 Proof for packing-type problem with arbitrary A matrix

2.5.3.1 Proof of Theorem 44

In this section, we use the same notation as that used in Section 2.5.1.1. So, let $\mathcal{J}$ be a partition of the index set of columns of A (that is [n]). Let $V = \{v_1, \dots, v_q\}$ be the vertices of $G^{\text{pack}}_{A,\mathcal{J}}$ (based on Definition 77). Let $\tilde{\mathcal{V}} = \{V^{u_1}, V^{u_2}, \dots, V^{u_k}\} \subseteq \mathcal{V}$ be the subset of sparse cut support list corresponding to the definition of corrected average density $D_{\mathcal{V}}$ (see Definition 43), so $V = \bigcup_{i=1}^k V^{u_i}$ and $\frac{1}{k}\sum_{i=1}^k |V^{u_i}| = D_{\mathcal{V}}$.

Recall from Section 2.5.1.1 a couple of definitions: first, the function ϕ maps subsets of vertices of V to the corresponding variable indices, namely if $S = \{v_i\}_{i\in I}$ then $\phi(S) = \bigcup_{i\in I} J_i$.

For the purpose of this section, let $x^{(S)} = \operatorname{argmax}\left\{(c|_{\phi(S)})^T(x|_{\phi(S)}) \mid x \in P^I\right\}$. (Notice that this is different from the definition used in Section 2.5.1.1.)

Lemma 73. *For any set $\tilde{V} \in \tilde{\mathcal{V}}$ we have $z^{\mathcal{V},P} \leq c^T x^{(\tilde{V})} + \sum_{v \in V \setminus \tilde{V}} c^T x^{(v)}$.*

Proof. Proof. Fix any $\tilde{V} \in \tilde{\mathcal{V}}$ and $S \subseteq \tilde{V}$. Let $x^* = \operatorname{argmax}\{c^T x \mid x \in P^{\mathcal{V},P}\}$ be an optimal solution corresponding to the optimization over $P^{\mathcal{V},P}$. Since $P^{\mathcal{V},C} = \bigcap_{\tilde{V}' \in \tilde{\mathcal{V}}} P^{(\tilde{V}')}$, we have that $x^* \in P^{(S)} \supseteq P^{(\tilde{V}')}$. From Observation 6 we then get $x^*|_{\phi(S)} \in P^I|_{\phi(S)}$.

Thus we get $(c|_{\phi(S)})^T(x^*|_{\phi(S)}) \leq (c|_{\phi(S)})^T(x^{(S)}|_{\phi(S)}) \leq c^T x^{(S)}$, where the first inequality follows from optimality of $x^{(S)}$ and the second inequality follows from non-negativity of c and $x^{(S)}$.

In particular, since $\tilde{\mathcal{V}}$ covers V, we can apply this to the any singleton $S = \{v\}$ and get $(c|_{\phi(v)})^T(x^*|_{\phi(v)}) \leq c^T x^{(v)}$.

Applying this bound, we obtain that for any $\tilde{V} \in \mathcal{V}$

$$z^{\mathcal{V},P} = c^T x^* = (c|_{\phi(\tilde{V})})^T(x^*|_{\phi(\tilde{V})}) + \sum_{v \in V \setminus \tilde{V}} (c|_{\phi(v)})^T(x^*|_{\phi(v)}) \leq c^T x^{(\tilde{V})} + \sum_{v \in V \setminus \tilde{V}} c^T x^{(v)}.$$

This concluses the proof. $\square$

Now we are ready to complete the proof of the theorem. Using Lemma 73 for all sets in $\tilde{\mathcal{V}}$ and adding up these inequalities we obtain that

$$\begin{aligned} k \cdot z^{\mathcal{V},P} &\leq \sum_{i=1}^{k} c^T x^{(V^{u_i})} + \sum_{i=1}^{k} \left(\sum_{v \in V \setminus V^{u_i}} c^T x^{(v)} \right) \\ &= \sum_{i=1}^{k} c^T x^{(V^{u_i})} + \sum_{v \in V} \mathsf{miss}(v) \cdot c^T x^{(v)}, \end{aligned} \tag{13}$$

where $\mathsf{miss}(v) = |\{i \in [k] \mid v \notin V^{u_i}\}|$, that is the number of sparse-cut types in $\mathcal{V}$ in which the variables corresponding to vertex v do not appear.

Moreover it follows from the definition of $x^{(S)}$ that $x^S \in P^I$ and therefore we have that $z^I \geq c^T x^{(S)}$ for every subset $S \subseteq V$. Thus, we obtain that

$$\begin{aligned} z^I &\geq \max\left\{\max_{i\in[k]}\{c^T x^{(V^{u_i})}\}\ ,\ \max_{v\in V}\{c^T x^{(v)}\}\right\} \\ &\geq \frac{1}{k+\sum_{v\in V}\mathsf{miss}(v)}\left(\sum_{i=1}^{k} c^T x^{(V^{u_i})} + \sum_{v\in V}\mathsf{miss}(v)\cdot c^T x^{(v)}\right) \\ &\geq \frac{k}{k+\sum_{v\in V}\mathsf{miss}(v)}\cdot z^{\mathcal{V},P}. \end{aligned}$$

where the second inequality follows from taking a weighted average, the third inequality follows from (13). Finally, the next lemma shows that $k+\sum_v \mathsf{miss}(v) = k+kq-kD_{\mathcal{V}}$, concluding the proof of the theorem.

Lemma 74. $kD_{\mathcal{V}} + \sum_{v\in V}\mathsf{miss}(v) = kq.$

Proof. Proof. We perform a simple double counting. Consider the $V/\{V^i\}_i$ incidence matrix $B \in \{0,1\}^{k\times q}$ defined as $B_{i,v} = 1$ if $v \in V^i$ and $B_{i,v} = 0$ if $v \notin V^i$. Using the definition of $D_{\mathcal{V}}$ we have:

$$kD_{\mathcal{V}} = |\{(i,v)\in[k]\times V \mid B_{i,v}\neq 0\}|\,. \tag{14}$$

On the other hand, from the definition of $\mathsf{miss}(v)$ we have that

$$\sum_{v\in V}\mathsf{miss}(v) = \sum_{v\in V}|\{i\in[k]\mid v\notin V^i\}| = |\{(i,v)\in[k]\times V\mid B_{i,v}=0\}|\,. \tag{15}$$

By (14) and (15), we have that $kD_{\mathcal{V}} + \sum_{v\in V}\mathsf{miss}(v) = kq$. This concludes the proof. □

2.5.3.2 Proof of Theorem 46

We consider the following integer program with $2K-1$ variables:

$$
\begin{aligned}
\max\ & x_K + \sum_{j=K+1}^{2K-1} x_j \\
s.t.\ & \sum_{i=1}^{K} x_i = 1 && (16)\\
& x_i + x_j \leq 2-\epsilon \qquad \forall i \in \{1,\dots,K-1\}, \quad \forall j \in \{K+1,\dots,2K-1\} \setminus \{K+i\} && (17)\\
& x_K + x_j \leq 2-\epsilon \qquad \forall j \in \{K+1,\dots,2K-1\} && (18)\\
& x \in \{0,1\}^{2K-1}.
\end{aligned}
$$

(We assume $\epsilon < \frac{K-1}{K}$.) Let P denote the integer set relative to this problem.

We consider the partition $\mathcal{J} = \{J_1,\dots,J_K\}$ of the columns given by $J_1 = \{1,\dots,K\}$, $J_i = \{K+i-1\}$ for $i \in 2,\dots,K$. Notice that the packing interaction graph $G^{\text{pack}}_{A,\mathcal{J}}$ for this program is a star on K nodes. Writing explicitly $G^{\text{pack}}_{A,\mathcal{J}} = (V,E)$ with $V = \{v_1,\dots,v_K\}$ and

$$E = \{(v_1,v_2),(v_1,v_3),(v_1,v_4),\dots,(v_1,v_K)\}.$$

Since we are in the context of the super sparse closure, we have the support list

$$\mathcal{V} = \{\{v_1\},\{v_2\},\dots,\{v_K\}\}.$$

We show the bound $z^{S.S.} \geq K \cdot z^I - \epsilon$, and start by lower bounding $z^{S.S.}$.

Lemma 75. $z^{S.S.} \geq K - \epsilon$

Proof. Proof. We claim that the point $\bar{x}$ given by $\bar{x}_j = \frac{\epsilon}{K-1}$ for all $j = \{1,\dots,K-1\}$, $\bar{x}_K = 1-\epsilon$, $\bar{x}_j = 1$ for all $j \in \{K+1,\dots,2K-1\}$ belongs to the natural sparse closure $P^{S.S.}$, proving it using Observation 6.

First, it is easy to check that $\bar{x}$ belongs to the LP relaxation P^{LP}. Moreover, note that $P^I|_{J_1} = \{(x_1,\dots,x_K) \in [0,1]^K \mid \sum_{i=1}^K x_i = 1\}$, and hence $\bar{x}|_{J_1} \in P^I|_{J_1}$. In addition, $P^I|_{J_j} = [0,\ 1]$ for $j \in \{2, \ldots, |V|\}$, and thus $\bar{x}|_{J_j} = \bar{x}_{K+j-1} \in P^I|_{J_j}$. Since $P^{S.S.} = \bigcap_{j=1}^{K} P^{(J_j)}$, from Observation 6 we get that $\bar{x} \in P^{S.S.}$. □

To complete the proof, we show that the optimal value of the IP is (at most) 1, namely exactly one of the variables $x_K, x_{K+1}, \ldots, x_{2K-1}$ can take a value of 1 and the others are zero. So consider any feasible solution $\bar{x} \in \{0,1\}^{2K-1}$. If $\bar{x}_K = 1$, then the constraints (18) imply that $x_{K+1} = x_{K+2} = \cdots = x_{2K-1} = 0$. On the other hand if $\bar{x}_K = 0$, then by constraint (16) there is some $i \in [K-1]$ with $\bar{x}_i = 1$, and so constraints (17) imply $\bar{x}_j = 0$ for all $j \in \{|V|+1, \ldots, 2K+1\} \setminus \{K+i\}$, and so at most $\bar{x}_j$ can take value 0.

Since $z^I \leq 1$ and $z^{S.S.} \geq K - \epsilon$, we get the desired bound $z^{S.S.} \geq K \cdot z^I - \epsilon$, concluding the proof of Theorem 46.

2.5.3.3 Proof of Theorem 48

We will construct an example with $2K$ variables. To start, for $k \in [K]$ let P_k^I be the convex hull of the points

$$\begin{aligned} P_k := \big\{ &(x,y) \in \{0,1\}^{K+K} \;\big| \\ &y_k = 1 \text{ if and only if either } [x_k = 1, x_i = 0\ \forall i \neq k] \text{ or } [x_k = 0, x_i = 1\ \forall i \neq k] \big\} \end{aligned}$$

We then consider the integer program

$$\begin{aligned} \max\ & \sum_{k=1}^{K} y_k \\ \text{s.t.}\ & (x,y) \in \bigcap_{k \in [K]} P_k^I \cap \{0,1\}^{2K}. \end{aligned}$$

Let P denote the associated integer set. The partition of variable indices we consider is $\mathcal{J} = \{J_0, J_1, J_3, \ldots, J_K\}$, where J_0 corresponds to the variables x, and each J_k corresponds to variable y_k for $k \in [K]$. Notice that the packing interaction graph $G^{\text{pack}}_{A,\mathcal{J}}$ is a star on $K+1$ nodes; explicitly, $G^{\text{pack}}_{A,\mathcal{J}} = (V,E)$ with $V = \{v_0, \ldots, v_K\}$ and

$E = \{\{v_0, v_1\}, \ldots, \{v_0, v_K\}\}$. Recall we are in the natural sparse closure setting, so the support list $\mathcal{V}$ in this case equals the edge set E.

We show that $z^{N.S.} \geq K \cdot z^I$. For that, we start by lower bounding $z^{N.S.}$.

Lemma 76. *$z^{N.S.} \geq K$.*

Proof. Proof. We show that the solution $(\bar{x}, \bar{y})$ given by $\bar{x} = \frac{1}{2}\mathbb{1}$ and $\bar{y} = \mathbb{1}$ belongs to $P^{N.S.}$. Following Observation 6, to show $(\bar{x}, \bar{y}) \in P^{N.S.}$ it suffices to show $(\bar{x}, \bar{y}) \in P^{LP}$ and $(\bar{x}, \bar{y}_k) \in P^I|_{(x,y_k)}$ for all $k \in [K]$. Notice that $P^{LP} = \bigcap_{k\in[K]} P^I_k$ and $P^I|_{(x,y_k)} = P^I_k|_{(x,y_k)}$ (the latter uses the fact $P^I_j|_{(x,y_k)} = [0,1]^{K+1}$ for $j \neq k$). Thus it suffices to show $(\bar{x}, \bar{y}) \in P^I_k$ for all $k \in [K]$

For that, fix $k \in [K]$ and consider the points (x^{k1}, e^k) and (x^{k2}, e^k), where e^i is the ith canonical basis vector and

$$x^{k1}_i = \begin{cases} 1 & \text{if } i = k, \\ 0 & \text{otherwise} \end{cases}$$

$$x^{k2}_i = \begin{cases} 0 & \text{if } i = k, \\ 1 & \text{otherwise} \end{cases}.$$

By definition both these points belong to P_k; the average $\frac{1}{2}(x^{k1}, e^k) + \frac{1}{2}(x^{k2}, e^k) = (\frac{1}{2}\mathbb{1}, e^k)$ also belongs to P_k. Moreover, since the constraints defining P^I_k are independent of variable y_i for $i \neq k$, we have that $(\frac{1}{2}\mathbb{1}, \mathbb{1}) = (\bar{x}, \bar{y})$ also belongs to P^I_k. This concludes the proof. $\square$

Now it is easy to see from the definition of P_k that no feasible solution to the IP can set more than one y variable to 1, and hence the optimal value z^i is at most 1. Together with the previous lemma, this gives the desired bound $z^{N.S.} \geq K \cdot z^I$, thus concluding the proof of the theorem.

Chapter III

APPROXIMATION ALGORITHM FOR SPARSE PACKING INTEGER PROGRAMS

3.1 Introduction

A packing integer program (IP) is a problem of the following form:

$$\begin{array}{lll}
\text{(P)} & \max & c^T x \\
 & s.t. & Ax \leq b \\
 & & x \in \mathbb{Z}^n_+,
\end{array}$$

where $A \in \mathbb{R}^{m \times n}_+$, $b \in \mathbb{R}^m_+$, and $c \in \mathbb{R}^n_+$. In this chapter, we present a new approximation algorithm for solving extremely large-scale packing IPs which exploits global sparsity pattern of the constraint matrix A which we assume is known in advance.

Our method to describe the global sparsity pattern of A is via a graph that is based on partitioning the columns of the matrix A into blocks that represents nodes of the graph and then generalizing the concept of intersection graph [20] to describe the edges of the graph. This is the same method of describing the sparsity pattern that we used in a recent study on the effectiveness of sparse cutting-planes for sparse packing IPs [18].

Our algorithm is essentially a decomposition heuristic with performance guarantees. It runs in two phases. In the first phase, the sparse packing problem is partitioned into smaller parts and then these small integer programs are solved. In the second phase, the optimal solutions of the smaller problems are patched together into a feasible solution for the original problem by exploiting the sparsity structure of the constraint matrix. Notice that by design this algorithm is highly parallelizable.

We present a theoretical guarantee on the quality of the solution produced by this algorithm, which we show depends only on the sizes of the smaller IPs and the sparsity structure, otherwise being completely data independent. Based on recent work on exploiting combinatorial implosion [24], we also propose to use a computational method to compute alternative dual bounds using the optimal values of smaller packing IPs.

Our algorithm can be viewed as a generalization of some ideas presented for two-stage stochastic matching IPs in the papers [28, 19]. At a high level, similar ideas have also been used to solve multi-stage capacity expansion problem [3]. We however note that our algorithm is much more general in scope (and particularly its analysis), especially since it can be used whenever the instance's global sparsity structure is known in advance (which may not be the same as that of stochastic programming instances). We mention in passing that a number of randomized LP-rounding based approximation algorithms for sparse packing IPs have also been proposed [5, 35]. Our algorithm, as discussed above, is quite different in spirit.

We experiment on randomly generated sparse instances and also packing integer programs with tree structure and compare the results of our algorithm to those by `CPLEX` and also a specialized heuristic for packing integer programs based on the state-of-the-art the Greedy Randomized Adaptive Search Procedure (GRASP) of [15]. The instances we test on are extremely large-scale and `CPLEX` performs quite poorly on these instances. Our results indicate that on these instances, our algorithm outperforms GRASP as well.

Outline of the chapter: In Section 3.2 we present our algorithm and in Section 3.3 we present the two methods to obtain dual bounds, one theoretical which can be precomputed and is instance independent while the other is computational that gives instance-specific bounds. In Section 3.4 we present results from our computational results. We make some concluding remarks in Section 3.5.

3.2 Algorithm

Given a positive integer integer n, we represent the set $\{1, \ldots, n\}$ by $[n]$.

First we present some necessary definitions in Section 3.2.1, then present the details of our algorithm in Section 3.2.2 and finally in Section 3.2.3 we show that the algorithm produces a feasible solution.

3.2.1 Some definitions

In this section, we introduce some definitions that are used throughout this chapter. These definitions were introduced in Chapter 2. But for the convenience of reading, we introduce the related definitions again in this chapter, which are useful in describing the algorithm.

Let $[n]$ be the index set of all columns and $[m]$ be the index set of all rows. Let $\mathcal{J} = \{J_1, J_2 \ldots, J_q\}$ be a partition of $[n]$ into blocks, that is, $J_i \subseteq [n]$ for all $i \in [q]$, $\bigcup_{i=1}^{q} J_i = [n]$ and $J_i \cap J_j = \emptyset$ when $i \neq j$. We call each element of $\mathcal{J}$ a block. We next define the packing interaction graph of a matrix A, which encodes the global sparsity structure of A.

Definition 77 (Packing interaction graph of A)**.** *Consider a matrix $A \in \mathbb{R}_+^{m \times n}$. Let $\mathcal{J} := \{J_1, J_2 \ldots, J_q\}$ be a partition of $[n]$. We define the* packing interaction graph *$G_{A,\mathcal{J}}^{pack} = (V, E)$ as follows:*

1. *There is one node $v_j \in V$ for every block $J_j \in \mathcal{J}$.*
2. *For all $v_i, v_j \in V$, there is an edge $(v_i, v_j) \in E$ if and only if there is a row of A with non-zero entries in both blocks J_i and J_j, namely there is $k \in [m]$, $u \in J_i$ and $w \in J_j$ such that $A_{ku} \neq 0$ and $A_{kw} \neq 0$.*

There is a natural identification of sets of nodes of $G_{A,\mathcal{J}}^{\text{pack}}$ with sets of indices of variables, namely if $\mathcal{J} = \{J_1, J_2, \ldots, J_q\}$ is the given variable index partition and the nodes of $G_{A,\mathcal{J}}^{\text{pack}}$ are $\{v_1, v_2, \ldots, v_q\}$, then the set of vertices $\{v_i\}_{i \in I}$ corresponds to the indices $\bigcup_{i\in I} J_i \subseteq [n]$. We will make use of this correspondence, and in order to make statements precise we use the function $\phi : 2^{V(G^{\text{pack}}_{A,\mathcal{J}})} \to 2^{[n]}$ to denote this correspondence; with slight abuse of notation, for a singleton set $\{v\}$ we use $\phi(v)$ instead of $\phi(\{v\})$. Similarly, for a collection of node subsets $\mathcal{C} = \{C_1, \ldots, C_s\}$ of $G^{\text{pack}}_{A,\mathcal{J}}$, with slight abuse of notation, we define $\phi(\mathcal{C}) = \bigcup_{C_i\in\mathcal{C}} \phi(C_i)$.

Next we introduce some graph-theoretical definitions.

Definition 78 (Support list $\mathcal{V}$ and particle collection $\mathcal{C}(\mathcal{V})$)**.** *Consider a simple graph $G = (V, E)$, and let $\mathcal{V}$ be a collection of subsets of V. We say $\mathcal{V}$ is a support list of V if it covers V, that is it satisfies the following condition:*

$$V \subseteq \bigcup_{V_i\in\mathcal{V}} V_i.$$

The particle collection of $\mathcal{V}$, denoted as $\mathcal{C}(\mathcal{V})$, is a collection of subsets of V which satisfies two conditions: (i) Each element of $\mathcal{C}(\mathcal{V})$ is a subset of an element of $\mathcal{V}$ and (ii) $\mathcal{C}(\mathcal{V})$ is closed under taking subsets. In other words,

$$\mathcal{C}(\mathcal{V}) = \{C \subseteq V \mid \exists V_i \in \mathcal{V}, C \subseteq V_i\}.$$

To simplify the notation, as long as the support list $\mathcal{V}$ is obvious, we use $\mathcal{C}$ to represent $\mathcal{C}(\mathcal{V})$.

In our algorithm we select a support list $\mathcal{V}$ and the IPs that we solve correspond to the particle list $\mathcal{C}$ of $\mathcal{V}$, that is, for every element $C \in \mathcal{C}$ we solve an IP with variables $\phi(C)$.

Given $x \in \mathbb{R}^n$ and $N \subseteq [n]$, we use $x|_N \in \mathbb{R}^{|N|}$ to denote the projection of x on the coordinates indexed by N and we use $\tilde{x}|_N \in \mathbb{R}^n$ to denote the extension of $x|_N$ such that the projection of $\tilde{x}|_N$ on N equals to $x|_N$ and rest of the entries are equal to 0.

3.2.2 Details of algorithm

As discussed before, our algorithm runs in two phases. In the first phase, we solve multiple sub-IPs derived from original problem (P). In the second phase, we utilize the graph structure $G^{\text{pack}}_{A,\mathcal{J}}$ as well as the information obtained from first phase to construct a feasible solution for (P).

Other than the packing instance, the inputs to our algorithm is the partition $\mathcal{J} := \{J_1, J_2 \ldots, J_q\}$ of the columns of A into blocks and a support list $\mathcal{V}$.

While the choice of $\mathcal{J}$ can be arbitrary, it must ideally be selected to reflect the sparsity structure of A, that is the resulting graph $G^{\text{pack}}_{A,\mathcal{J}}$ should be sparse. For example, a natural choice in the case of a multi-stage stochastic program, it to select all the variables corresponding to a particular realization in a particular stage as one block of variables.

As previously discussed, the sub-IPs solved in phase 1 correspond to the elements in the particle collection of a support list $\mathcal{V}$. Therefore the choice of $\mathcal{V}$ must be guided with the need to balance the need to improve the expected solution quality (which typically improves with a bigger size of $\mathcal{C}$) and tractability is terms of the number of sub-IPs to solve. We discuss later the choice of $\mathcal{V}$ in our computations.

We first present the pre-processing steps as Algorithm 1.

Algorithm 1 Pre-processing for approximation algorithm

input: $A \in \mathbb{R}^{m\times n}_{+}, \mathcal{J} = \{J_1, \ldots, J_q\}$, $\mathcal{V}$.
Construct $G^{\text{pack}}_{A,\mathcal{J}}(V, E)$.
Construct $\mathcal{C}$ based on $\mathcal{V}$

Algorithm 2 is the (outline of the) main algorithm. In phase 1, we solve multiple sub-IPs for corresponding optimal solutions $x^1, \ldots, x^s$ and optimal objective function values $w_1, \ldots, w_s$. This is obtained by calling sub-routine Algorithm 3. Then in the second phase we construct a feasible solution of the original packing problem by calling sub-routine Algorithm 4.

Algorithm 2 Approximation algorithm
$(x^*, obj) = APP(A, b, c, \mathcal{J}, G^{\text{pack}}_{A,\mathcal{J}}(V, E), \mathcal{V}, \mathcal{C})$

input: $A \in \mathbb{R}^{m\times n}_+$, $b \in \mathbb{R}^m_+$, $c \in \mathbb{R}^n_+$, $\mathcal{J}$, $\mathcal{V}$, $\mathcal{C} = \{C_1, \ldots, C_s\}$, $G^{\text{pack}}_{A,\mathcal{J}}(V, E)$.
(Phase 1)
for $i = 1$ to s **do**
  Solve $(x^i, w_i) = PART(A, b, c, \mathcal{J}, C_i)$ calling Algorithm 3, $\forall i \in [s]$.
end for
(Phase 2)
Solve $(x^*) = MSS(G^{\text{pack}}_{A,\mathcal{J}}(V, E), \mathcal{V}, \mathcal{C}, X, W)$ calling Algorithm 4
return $(x^*,\ c^T x^*)$

Given a set of vertices C_i, Algorithm 3 involves solving the 'projection of (P) on $\phi(C_i)$ variables', that is, all the variables in $[n] \setminus \phi(C_i)$ are fixed to 0.

Algorithm 3 Solving partial optimization $(x^i, w_i) = PART(A, b, c, \mathcal{J}, C_i)$

input: $A \in \mathbb{R}^{m\times n}_+$, $b \in \mathbb{R}^m_+$, $c \in \mathbb{R}^n_+$, $\mathcal{J} = \{J_1, \ldots, J_q\}$, $C_i \in \mathcal{C}$.
Solve the following IP:

$$
\begin{array}{lll}
(P|_{C_i}) & \max & c^T x \\
& & Ax \leq b \\
& & x|_{\phi(V\setminus C_i)} = 0 \\
& & x \in \mathbb{Z}^n_+
\end{array}
$$

Let x^i be the optimal solution and $w_i = c^T x^i$.
return $(x^i,\ w_i)$

Algorithm 4 describes the patching algorithm of phase 2. The patching to obtain a feasible is obtained by solving a IP called as (MS).

3.2.3 Correctness of algorithm

Given a packing problem (P) with $A \in \mathbb{R}^{m\times n}_+, b \in \mathbb{R}^m_+$ and $c \in \mathbb{R}^n_+$ as well as a selection $\mathcal{J}$ and $\mathcal{V}$, let x^* be the optimal solution obtained by Algorithm 2. We show the correctness as below.

Theorem 79 (Correctness of algorithm)**.** *Let x^* be the solution produced by Algorithm 2 with column partition $\mathcal{J}$ and support list $\mathcal{V}$. Then x^* is a feasible solution of* (P)*.*

Algorithm 4 Solving maximal combination $(x^*) = MSS(G^{\text{pack}}_{A,\mathcal{J}}(V,E), \mathcal{V}, \mathcal{C}, X, W)$

input: $G^{\text{pack}}_{A,\mathcal{J}}(V,E)$, $X = \{x^1, \dots, x^s\}$, $\mathcal{V} = \{V_1, \dots, V_k\}$, $\mathcal{C} = \{C_1, \dots, C_s\}$, $W = \{w_1, \dots, w_s\}$.

Solve the following IP:

$$
\begin{aligned}
\text{(MS)} \quad \max \quad & \sum_{i=1}^{s} w_i y_i \\
& y_i + y_j \leq 1, \forall i, j \in [s],\ s.t.\ C_i \bigcap C_j \neq \emptyset \\
& y_i + y_j \leq 1, \forall i, j \in [s],\ s.t.\ \exists v_1 \in C_i, v_2 \in C_j, (v_1, v_2) \in E \\
& y \in \{0,1\}^s
\end{aligned}
$$

Let y^* be the optimal solution and $x^* = \sum_{i=1}^{s} y_i x^i$.

return (x^*)

Proof. Let $x^1, \dots, x^s$ (where $s = |\mathcal{C}|$) be the solutions generated by Algorithm 3 and let y^* be the optimal solution obtained by Algorithm 4. With out loss of generality, assume that $y^*_j = 1$, $j = 1, \dots, u$ and $y^* = 0$, $j = u+1, \dots, s$. Let A_{i*} be the i^{th} row of A, and $S^i = \{j | A_{i*} * x^j \neq 0, \forall j \in [u]\}$, for all $i \in [m]$.

We first verify that $|S^i| \leq 1$ for all $i \in [m]$. If $u = 1$, then there is nothing to prove. Therefore assume $u \geq 2$ and assume by contradiction (and with out loss of generality) that $1, 2 \in S^1$, which implies that $A_{1*} * x^1 \neq 0$ and $A_{1*} * x^2 \neq 0$. Therefore there exists $v_1 \in C_1$ and $k_1 \in \phi(v_1)$, such that $A_{1k_1} \neq 0$. Similarly there exists $v_2 \in C_2$ and $k_2 \in \phi(v_2)$ such that $A_{1k_2} \neq 0$. Moreover note that the constraints defining (MS) together with the fact that $y^*_1 = y^*_2 = 1$ imply that $C_1 \cap C_2 = \emptyset$. Hence v_1 and v_2 are distinct nodes in $G^{\text{pack}}_{A,\mathcal{J}}$. Now, according to the definition of packing interaction graph, $A_{1k_1} \neq 0$ and $A_{1k_2} \neq 0$ implies that $(v_1, v_2) \in E$. Thus $y_1 + y_2 \leq 1$ is a constraint of (MS) in Algorithm 4, which is a contradiction to $y^*_1 = y^*_2 = 1$.

Then, for all row $i \in [m]$, we obtain that, $A_{i*} * x^* = A_{i*} * (\sum_{l=1}^{u} x^l) = A_{i*} * x^{u'} \leq b_i$ where $u' \in [u]$ and the last inequality follows from the fact that $x^{u'}$ is a feasible solution for $(P|_{C_{u'}})$ and therefore a feasible solution for (P). Thus $Ax^* \leq b$ and x^* is a feasible solution for (P). $\square$

3.3 Performance guarantees

First we present some necessary definitions in Section 3.3.1, then present details of a data-independent dual bound (that depends only on $G^{\text{pack}}_{A,\mathcal{J}}$ and choice of $\mathcal{V}$) that guarantees the performance of the algorithm in Section 3.3.2 and finally in Section 3.3.3 we describe how bounds may be obtained in a computationally tractable fashion.

3.3.1 Some definitions

The definitions presented in this section are generalizations of standard graph-theoretic notions. These definitions were presented in Chapter 2. For a similar reason as above, we re-introduce the definitions here.

Definition 80 (Mixed stable set subordinate to $\mathcal{V}$)**.** *Let $G = (V, E)$ be a simple graph. Let $\mathcal{V}$ be a collection of subsets of the vertices V. We call a collection of subsets of vertices $\mathcal{M} \subseteq 2^V$ a* mixed stable set subordinate to $\mathcal{V}$ *if the following hold:*

1. *Every set in $\mathcal{M}$ is contained in a set in $\mathcal{V}$*
2. *The sets in $\mathcal{M}$ are pairwise disjoint*
3. *There are no edges of G with endpoints in distinct sets in $\mathcal{M}$.*

The next definition generalizes the notion of fractional chromatic number of a graph.

Definition 81 (Fractional mixed chromatic number with respect to $\mathcal{V}$)**.** *Consider a simple graph $G = (V, E)$ and a collection $\mathcal{V}$ of subset of vertices. Given a mixed stable set $\mathcal{M}$ subordinate to $\mathcal{V}$, let $\chi_{\mathcal{M}} \in \{0, 1\}^{|V|}$ denote its incidence vector (that is, for each vertex $v \in V$, $\chi_{\mathcal{M}}(v) = 1$ if v belongs to a set in $\mathcal{M}$, and $\chi_{\mathcal{M}}(v) = 0$ otherwise.)*

Then we define the fractional mixed chromatic number $\eta^{\mathcal{V}}(G)$ *as:*

$$\begin{aligned} \eta^{\mathcal{V}}(G) = \ \min \quad & \textstyle\sum_{\mathcal{M}} y_{\mathcal{M}} \\ \text{s.t.} \quad & \textstyle\sum_{\mathcal{M}} y_{\mathcal{M}} \chi_{\mathcal{M}} \geq \mathbb{1} \\ & y_{\mathcal{M}} \geq 0 \quad \forall \mathcal{M}, \end{aligned}$$

where the summations range over all mixed stable sets subordinate to $\mathcal{V}$ *and* $\mathbb{1}$ *is the vector in* $\mathbb{R}^{|V|}$ *of all ones.*

3.3.2 Data-independent guarantee

In this section, we prove the following result.

Theorem 82. *Let* x^P *be the optimal solution of* (P)*, and* x^* *be the solution produced by Algorithm 2 with column partition* $\mathcal{J}$ *(resulting in packing interaction graph* $G^{pack}_{A,\mathcal{J}}$*) and support list* $\mathcal{V}$*. Then* $c^T x^* \geq \frac{c^T x^P}{\eta^{\mathcal{V}}(G^{pack}_{A,\mathcal{J}})}$.

Theorem 82 shows that our algorithm is a $\eta^{\mathcal{V}}(G^{\text{pack}}_{A,\mathcal{J}})$-approximation algorithm. To prove the theorem, we first prove the following result.

Lemma 83. *Let* $\mathcal{M}$ *be a mixed stable set subordinate to* $\mathcal{V}$*. Then for all* x *feasible for* (P)*,* $c^T(\tilde{x}|_{\phi(\mathcal{M})}) \leq c^T x^*$.

Proof. Assume x^0 is an arbitrary feasible solution for (P). We will show that $c^T(\tilde{x}^0|_{\phi(\mathcal{M})}) \leq c^T x^*$. Recall the notation that $|\mathcal{C}| = s$. Let $\hat{w} = (\hat{w}_1, \ldots, \hat{w}_s)$ where $\hat{w}_i$ is obtained from Algorithm 3 for all i. Let $y^* \in \{0,1\}^s$ be the optimal solution obtained from Algorithm 4.

Observe that:

$$c^T x^* = c^T \left(\sum_{i=1}^{s} y_i^* x^i \right) = \sum_{i=1}^{s} y_i^* \left(c^T x^i \right) = \sum_{i=1}^{s} y_i^* \hat{w}_i = \hat{w}^T y^*.$$

Observe that the set of constraints of (MS) in Algorithm 4 encode mixed stable sets subordinate to $\mathcal{V}$. Thus defining $y^1 \in \{0,1\}^s$ as

$$y_i^1 = \begin{cases} 1 & \text{if } C_i \in \mathcal{M} \\ 0 & \text{if } C_i \notin \mathcal{M}, \end{cases} \tag{19}$$

we have that y^1 satisfies the constraints of (MS).

Then construct $\bar{w} = (\bar{w}_1, \ldots, \bar{w}_s)$ where $\bar{w}_i = c^T(\tilde{x}^0|_{\phi(C_i)})$ for all i. Since x^0 is feasible for (P), $\tilde{x}^0|_{\phi(C_i)}$ is feasible for $(P|_{\phi(C_i)})$. Thus we obtain that $\bar{w}_i \leq \hat{w}_i$.

By reorganizing the solution, we have

$$\begin{aligned} c^T(\tilde{x}^0|_{\phi(\mathcal{M})}) &= c^T\left(\sum_{C_i \in \mathcal{M}} \tilde{x}^0|_{\phi(C_i)}\right) \\ &= \sum_{C_i \in \mathcal{M}} \bar{w}_i \\ &= \sum_{i=1}^{s} y_i^1 \bar{w}_i \\ &\leq \sum_{i=1}^{s} y_i^1 \hat{w}^i \\ &\leq \sum_{i=1}^{s} y_i^* \hat{w}^i = c^T x^*, \end{aligned}$$

where the last inequality follows from the construction of (MS). □

Now we prove Theorem 82.

Proof of Theorem 82. For simplicity of notation let $G := G^{\text{pack}}_{A,\mathcal{J}}$. Let $\mathcal{W} = \{\mathcal{M}_1, \ldots, \mathcal{M}_t\}$ be the collection of all mixed stable sets subordinate to $\mathcal{V}$. According to the definition of fractional mixed chromatic number we have that

$$\begin{aligned} \eta^{\mathcal{V}}(G) = \quad \min \quad & \sum_{i=1}^{t} u_i \\ \text{s.t.} \quad & \sum_{i=1}^{t} u_i \chi_i \geq \mathbb{1} \\ & u \geq 0. \end{aligned}$$

Let u^* be the optimal solution of the above linear program, that is, $\eta^{\mathcal{V}}(G) = \sum_{i=1}^t u_i^*$. Define $g = \sum_{i=1}^t u_i^* \chi^i \in \mathbb{R}^q$. Then we have

$$\begin{aligned}
c^T x^P &= \sum_{j=1}^{q} (c|_{\phi(v_j)})^T (x^P|_{\phi(v_j)}) \\
&\leq \sum_{j=1}^{q} (g_j * (c|_{\phi(v_j)})^T (x^P|_{\phi(v_j)})) \\
&= \sum_{j=1}^{q} \left(\left(\sum_{i=1}^{t} u_i^* \chi^i \right)_j * (c|_{\phi(v_j)})^T (x^P|_{\phi(v_j)}) \right) \\
&= \sum_{j=1}^{q} \left(\left(\sum_{i=1}^{t} u_i^* \chi_j^i \right) * (c|_{\phi(v_j)})^T (x^P|_{\phi(v_j)}) \right) \\
&= \sum_{i=1}^{t} u_i^* \left(\sum_{j=1}^{q} (\chi_j^i * (c|_{\phi(v_j)})^T (x^P|_{\phi(v_j)})) \right) \\
&= \sum_{i=1}^{t} u_i^* \left(\sum_{v_j \in \mathcal{M}_i} (c|_{\phi(v_j)})^T (x^P|_{\phi(v_j)}) \right) \\
&= \sum_{i=1}^{t} u_i^* (c|_{\phi(\mathcal{M}_i)})^T (x^P|_{\phi(\mathcal{M}_i)}) \\
&= \sum_{i=1}^{t} u_i^* c^T \tilde{x}^P|_{\phi(\mathcal{M}_i)} \\
&\leq \sum_{i=1}^{t} u_i^* c^T x^* \\
&= \eta^{\mathcal{V}}(G) c^T x^*,
\end{aligned}$$

where the last inequality follows from Lemma 83. $\square$

3.3.3 Data-dependent computational upper bound

In order to estimate the performance of solution produced by Algorithm 2, we require a valid (dual) upper bound for P. For very large-scale problems, even solving the linear programming relaxation can be computationally expensive, as we observed in preliminary experiments.

Fortunately, it turns out, one can use the information $W = \{w_1, \ldots, w_s\}$ collected in Algorithm 3 to generate dual bounds using a technique from [24]. We describe this procedure in Algorithm 5.

Algorithm 5 Computational upper bound $(z^{UB}) = CUB(c, \mathcal{C}, W)$

---

input: $c \in \mathbb{Q}^n$, $\mathcal{C} = \{C_1, \ldots, C_s\}$, $W = \{w_1, \ldots, w_s\}$.
Solve the following LP:

$$
\begin{aligned}
\text{(CUB)} \quad z^{UB} = \max \quad & c^T x \\
s.t. \quad & \sum_{j \in \phi(C_i)} c_j x_j \leq w_i, \forall i \in [l] \\
& 0 \leq x \leq \mathbb{1}
\end{aligned}
$$

return (z^{UB})

Notice that by definition of the w_is (see Algorithm 3), any feasible solution $\hat{x}$ of (P), is also a feasible solution of (CUB) and therefore z^{UB} is a valid upper bound. We refer the reader to [24] to see a formal proof of validity of z^{UB}.

3.4 Computational experiments

In this section, we examine the performance of the proposed approximation algorithm in solving two classes of *set packing problems*. We remind the reader that set packing instances are of the form:

$$
\begin{aligned}
\text{(SP)} \quad \max \quad & c^T x \\
s.t. \quad & Ax \leq \mathbb{1} \\
& x \in \{0,1\}^n,
\end{aligned}
$$

where $A \in \{0,1\}^{m \times n}$, $c \in \mathbb{Q}^n_+$.

3.4.1 GRASP

The basic idea of GRASP is that it starts with a feasible solution (default starting point is $\mathbf{0}$) and then tries to randomly add more items to the solution. We will compare the performance of Algorithm 2 against the GRASP heuristic, which is one of the state-of-the-art methods to solve packing IPs [15]. We will also implemented a heuristic combining our approximation algorithm with GRASP.

Broadly speaking the GRASP implementation for set packing IPs [15] involves five features: (i) A random greedy procedure to build a pool of solutions, (ii) a so-called 'learning feature' that changes the way the variables are evaluated based on previously found solutions, (iii) a so-called 'reactive feature' that updates the level of randomization in the GRASP algorithm based on previously found solutions, (iv) a local search method to improve the quality of solution, and (v) a *path-linking* method to also discover better solutions using a pool of solutions.

We also did not implement path-linking since as shown in [15], this step is significantly more time consuming than other steps but produces the least improvement in the quality of solutions. We do not report results using the reactive[1] and learning[2] feature, since preliminary experiments showed no advantage of using these features for our instances. We present a brief description of GRASP and local search in Appendix C. We describe all the choices we make in the next section.

3.4.2 Software and hardware

We implement the algorithms using `CPLEX` callable libraries (version 12.5) and run the algorithms on Intel Xeon E5520 8 core Linux servers installed with 48 Gb RAM. Every program is run using single-threading (see discussion in Section 3.5).

3.4.3 Instance generation

We test two different kinds of instances: (i) 'random graph' based instances, that is, instances where we first generate a random graph G and then constructs the A matrix randomly so that with a suitable selection of $\mathcal{J}$, we obtain that $G^{\text{pack}}_{A,\mathcal{J}} = G$, (ii) instances whose packing interaction graph is a tree with three layers. These instances include but not limited to the three-stage stochastic programming instances with

[1] Reactive feature makes extensive use of local-search and as we discuss later and also show numerically, our instances are too large for running local search often.

[2] Learning feature usually produced results that were very similar to those produced by GRASP; and in fact most often slightly worse than solutions produced by GRASP for our instances.

Markov property. Since we generate set packing instances, we only need to describe how that objective function and constraint matrix is generated. We first present how we generate constraint matrix given a certain underlying graph $G(V, E)$. We apply this for both classes of instances.

Given a target packing interaction graph, say $G(V, E)$ (where $\texttt{nv} = |V|$), we construct a matrix that can be partitioned into $|E| \times |V|$ blocks with each block of size $\texttt{sqr} \times \texttt{sqr}$ (where $\texttt{sqr}$ is a parameter). Thus the constraint matrix has $|E| \times \texttt{sqr}$ rows and $|V| \times \texttt{sqr}$ columns. The $(i, j)^{\text{th}}$ block is all zeros if edge i is not incident to node j. Else if edge i is incident to node j, then the $(i, j)^{\text{th}}$ block is a randomly generated sparse matrix: we assign each entry the distribution of $Bernoulli(p_{nz})$, where $\texttt{p}_{\texttt{nz}}$ is a parameter.

3.4.3.1 Random packing interaction graph based instances

We generate a random *Erdös-Rényi* graph as our target packing interaction graph with $\texttt{nv}$ vertices (where $\texttt{nv}$ is a parameter), that is, given two nodes there is an edge connecting them with probability $\texttt{p}_{\texttt{e}}$ (where $\texttt{p}_{\texttt{e}}$ is a parameter). Since a disconnected graph implies that the set packing problem is decomposable, we only accept graphs that are connected.

For the objective function, each entry follows a distribution of $uniform\{1, \ldots, \texttt{M}_{\texttt{obj}}\}$, where $\texttt{M}_{\texttt{obj}}$ is a parameter.

Parameter values and number of instances

1. There are 20,000 variables in total. Three possible pairs of $(\texttt{nv}, \texttt{sqr})$ used are $(50, 400)$, $(100, 200)$, and $(125, 160)$.

2. Given the number of vertices $\texttt{nv}$, there are four choices of $\texttt{p}_{\texttt{e}}$: $\{3\%, 5\%, 8\%, 10\%\}$. Three graphs are generated for each $\texttt{p}_{\texttt{e}}$.

3. In each block, there are four choices of $\mathtt{p_{nz}}$: $\{5\%, 10\%, 15\%, 20\%\}$. Five instances are generated for each $\mathtt{p_{nz}}$.

4. $\mathtt{M_{obj}} = 50$.

3.4.3.2 Tree-structured instances

In this section we test the instances based on a tree with three layers. Since a two-stage stochastic program triggers a tree with two layers as the packing interaction graph, it is intuitive to expend the tree in number of layers. It is notable that some of the three-stage stochastic programs has such a structure (as long as the first stage scenario is independent with the third stage ones). But it is not necessarily that all the three-stage stochastic programs share this structure.

Now we introduce the parameters to generate the graph $G^{\text{pack}}_{A,\mathcal{J}}$ and the instances. Since $G^{\text{pack}}_{A,\mathcal{J}}$ is a tree with three layers, we only need to decide the number of leaves in second and third layer. Given the root node v^0 (the first layer), we introduce $\mathtt{n_{ss}}$ vertices $(v_1, \ldots, v_{\mathtt{n_{ss}}})$ connecting v_0 in the tree. Then for each node $v_1, \ldots, v_{\mathtt{n_{ss}}}$ on the second layer, there are $\mathtt{n_{ts}}$ leaves connected to it. Thus, we construct a 3-layer tree that has $\mathtt{n_{ss}}$ nodes on second layer and $\mathtt{n_{ss}} * \mathtt{n_{ts}}$ leaves.

With respect to the objective function, first we set $c_j = 1$ for all variable j in the first stage. For all variable x_j that belongs to a node on second layer, $c_j = 1/\mathtt{n_{ss}}$. For all variable x_j that shows up in the blocks corresponding to a leave, $c_j = 1/(\mathtt{n_{ss}} * \mathtt{n_{ts}})$.

Parameter values and number of instances

1. There are two choices of $\mathtt{n_{ss}}$: $\{8, 10\}$.

2. There are three choices of $\mathtt{n_{ts}}$: $\{6, 8, 10\}$.

3. There are three choices of $\mathtt{sqr}$: $\{100, 150, 150\}$.

4. For every instance, in each block there are four choices of $\mathtt{p_{nz}}$: $\{5\%, 10\%, 15\%, 20\%\}$. Five instances are generated for each $\mathtt{p_{nz}}$.

3.4.4 Design of computational experiments

Rationale for the experiments we conduct: Given an instance, to conduct a fair comparison we must give each method equal time. Our basic approximation algorithm has a fixed running time, but GRASP needs a time or iteration cap. Therefore, we first ran our algorithm (and its variants) and used this time as a baseline to run CPLEX and GRASP (and its variants) for the same amount of time. In fact, for all other methods we gave 5% more time so as to err on the conservative side in evaluating our algorithm. Moreover since local search is expensive, we used a different time bench mark when evaluating them.

Following are the details of the steps in our experiments for a given instance.

Approximation algorithm In all our experiments, we selected $\mathcal{V} = \{\{v_i, v_j\} \mid (v_i, v_j) \in E\}$. Therefore, the particle collection is $\mathcal{C}(\mathcal{V}) = \{\{v_i\} \mid v_i \in V\} \bigcup \{\{v_i, v_j\} \mid (v_i, v_j) \in E\}$. We followed the following steps:

- **Basic approximation algorithm:** We implemented Algorithm 2, that is we obtain $(x^*, z^A) = APP(A, \mathbb{1}, c, \mathcal{J}, G^{\text{pack}}_{A,\mathcal{J}}, \mathcal{V}, \mathcal{C})$, where z^A represents the optimal value with approximation algorithm. Let $\mathtt{T^A_{AGS}}$ be the total running time for Algorithm 2.

- **Approximation algorithm with basic GRASP (without local search):** We attempt to improve x^* by applying basic version of GRASP, that is Algorithm 8, $(\hat{x}^*, z^{AG}) = GRASP(x^*, A, c, alphaSet, proba)$ (see below for our choices of *alphaSet*, *proba*). The stopping criteria is 260 iterations. This number of iterations is based on recommendations in [15]. Let $\mathtt{T^G_{AGS}}$ be the total running time Algorithm 8 and therefore $\mathtt{T^{AG}_{AGS}} := \mathtt{T^A_{AGS}} + \mathtt{T^G_{AGS}}$, is the total time for Algorithm 2 and Algorithm 8. $\mathtt{T}^{\mathtt{AG}}_{\mathtt{AGS}}$ will be our baseline time for comparison with other methods that do not use local search.

- **Approximation algorithm with GRASP and local search:** We implemented local search Algorithm 9 to obtain $(x^S) = LS(\hat{x}^*, A, c)$. Let $z^{AGS} = c^T x^S$ be the best value after applying local search. Let $\mathtt{T}^{\mathtt{S}}_{\mathtt{AGS}}$ be the total running time Algorithm 9 and therefore $\mathtt{T}^{\mathtt{AGS}}_{\mathtt{AGS}} := \mathtt{T}^{\mathtt{AG}}_{\mathtt{AGS}} + \mathtt{T}^{\mathtt{S}}_{\mathtt{AGS}}$, is the total time for Algorithm 2, Algorithm 8 and Algorithm 9. $\mathtt{T}^{\mathtt{AGS}}_{\mathtt{AGS}}$ will be our baseline time for comparison with other methods that use local search.

We also implemented Algorithm 5, which uses W obtained from approximation algorithm, to obtain the computational dual bound $(LP^{MSS}) = CUB(c, \mathcal{C}, W)$.

CPLEX Given the instance with information A, c and time T, we run `CPLEX` to solve the IP with time limit $1.05 * \mathtt{T}^{\mathtt{AG}}_{\mathtt{AGS}}$. We set `CPLEX`'s 'MIPEMPHASIS' parameter set to 'FEASIBILITY' so that `CPLEX` will frequently generate more feasible solutions as it optimizes the problem, at some sacrifice in the speed to the proof of optimality. Let $IP^{\mathtt{CPLEX}}$ be the optimal objective value `CPLEX` obtains and $LP^{\mathtt{CPLEX}}$ be the best bound it finds.

Variants of GRASP With respect to GRASP, the *alphaset* we used is

$$alphaset = \{0, 0.15, 0.3, 0.45, 0.5, 0.6, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 1\}.$$

Since $|alphaset| = 13$, the probability set *proba* includes 13 elements that all of them equals to 1/13. This setting is the same as that recommended in Section 5 of [15]. Given the instance with information A, c and time T used by our algorithm, we implement GRASP in the following steps.

- **Basic GRASP:** We implemented $(x_G, z^G) = GRASP(\vec{0}, A, c, alphaSet, proba)$ with time limit $1.05 * \mathtt{T}^{\mathtt{AG}}_{\mathtt{AGS}}$ as stopping criteria. Let z^G denote the objective function value of the best solution found by GRASP. Formally, we record the running time of this algorithm as $\mathtt{T}_{\mathtt{G}}^{\mathtt{G}}$. We record the number of iterations of GRASP as NIt_G^G.

- **GRASP with local-search:** Let $x_{new} = \vec{0}$ and $x_{old} = \vec{0}$. Let $z^{GS} = 0$. We run iterations of $GRASP(\vec{0}, A, c, alphaSet, proba)$ and pause the algorithm every 260 iterations and record the best solution x_{new} till that time point. If x_{new} is better than the previous best solution x_{old}, then update $x_{old} = x_{new}$ and apply local search Algorithm 9 on x_{new}: $(x^S) = LS(x_{new}, A, c)$. If $c^T x^S > z^{GS}$, update $z^{GS} = c^T x^S$. Return z^{GS} as the best result achieved by GRASP and local search.

  We terminate when the total running time exceeds the time limit $1.05 * \mathtt{T}_{\mathtt{AGS}}^{\mathtt{AGS}}$. We run at least one iteration of local search (preceded by 260 iterations of basic GRASP iterations), and in some cases this time may significantly exceed $1.05 * \mathtt{T}_{\mathtt{AGS}}^{\mathtt{AGS}}$. Therefore, we will report the time for GRASP with local-search explicitly for our experiments as $\mathtt{T}_{\mathtt{G}}^{\mathtt{GS}} + \mathtt{T}_{\mathtt{S}}^{\mathtt{GS}}$ where the first term is the (basic) GRASP running time and the second term is the local-search running time. We record the the number of iterations of basic GRASP as NIt_{GS}^G and the number of iterations of local-search as NIt_{GS}^S.

We note here that we have implemented local search much more sparingly than recommended in [15]. In particular, we choose to run local search on good solutions rather than all solutions. The reason for this is the following: Given the large sizes of our instances, local search is very time consuming, so we want to say “time budget” for rest of GRASP. Moreover, since local search continues to loop until improvements are made, we expect it to become very expensive if we run them starting from bad solutions (we see evidence of this in our experiments). Therefore (as discussed above), together with our approximation algorithm, we only apply local search on the best solution. Together with pure GRASP, we apply the local search on multiple "good solutions" (as explained above on x_{new} – instead of all solutions as recommended in [15]).

3.4.5 Results

3.4.5.1 Notations for tables

We use the following acronyms for different methods/features.

1. `A`: our approximation algorithm.
2. `G`: (basic) GRASP algorithm.
3. `L`: the learning feature of GRASP.
4. `S`: the local search feature of GRASP.
5. `B`: stands for $BestBound = min\{LP^{MSS}, LP^{\texttt{CPLEX}}, LP^{RELAX}\}$. We discovered that for almost all the instances, $BestBound = LP^{MSS}$.
6. `IP` stands for the value of $IP^{\texttt{CPLEX}}$.

A combination of letters implies that the corresponding methods/features have been used together. Also

1. To simplify the notation of z^A, we just use the letter `A` to represent it.
2. `N()` represents the number of instances that satisfies the condition in bracket. For example, `N(AG` $>$ `A)` means the number of instances satisfying $z^{AG} > z^A$.
3. `M()` represents the geometric mean of the value in bracket. For example, `M(B/A)` stands for the geometric mean of z^B/z^A.
4. $\texttt{GAP}^{\texttt{method}}$ represents the geometric mean of the relative gap with respect to the given method. The relative gap is calculated as $\frac{BestBound - z^{method}}{BestBound} \times 100\%$.

5. To simplify notation, $\eta(\texttt{G})$ is the fractional mixed chromatic number of the graph $G^{\text{pack}}_{A,\mathcal{J}}$ with respect to our choice of $\mathcal{V}$. If we do not have the exact value of fractional mixed chromatic number, then we report a lower bound on it, denoted as LB $\eta(\texttt{G})$. We estimate the lower bound on $\eta(G)$ using the following method: By solving an IP (similar to the one solved in Algorithm 4) we find a mixed stable set subordinate to $\mathcal{V}$ containing the largest number of vertices. Let the number of vertices in this mixed stable set be $MCard$. Then it is easy to verify that $|V|/MCard$ provides a lower bound for $\eta(G)$.

6. In the instance feature column, we describe the features of instances such as the number of nodes in $G^{\text{pack}}_{A,\mathcal{J}}$ as nv followed with its value. The phrase `ID` represents the index of the graph/instance. The column `NIns` displays the number of instances based on a certain graph $G^{\text{pack}}_{A,\mathcal{J}}$.

3.4.5.2 Results for random graph based instances

In this section, we show the results for random graph based instances. We generated multiple graphs for our experiments. However, based on the results, there is no significant difference in the trends between the different graphs with same features. Thus, in order to simplify the presentation, we present results for one graph for each feature.

Approximate algorithm's actual performance vs theoretical worst case performance In Table 7, we compare the value of $\frac{BestBound}{z^A}$ against the lower bound on fractional mixed chromatic number. The values of $\frac{BestBound}{z^A}$ are lower than the lower bounds on the fractional mixed chromatic number with respect to $\mathcal{V}$, indicting that the actual performance of the algorithm is significantly better than that of the worst case performance guarantee.

Table 7: $\frac{BestBound}{z^A}$ vs fractional mixed chromatic number (random graph instances)

Instance Features	NIns	M(B/A)	LB η(G)
nv50pe3sqr400ID1/pnz5	5	1.10	1.52
nv50pe3sqr400ID1/pnz10	5	1.09	1.52
nv50pe3sqr400ID1/pnz15	5	1.10	1.52
nv50pe3sqr400ID1/pnz20	5	1.00	1.52
nv50pe5sqr400ID1/pnz5	5	1.16	1.56
nv50pe5sqr400ID1/pnz10	5	1.18	1.56
nv50pe5sqr400ID1/pnz15	5	1.23	1.56
nv50pe5sqr400ID1/pnz20	5	1.00	1.56
nv50pe8sqr400ID1/pnz5	5	1.17	1.67
nv50pe8sqr400ID1/pnz10	5	1.20	1.67
nv50pe8sqr400ID1/pnz15	5	1.30	1.67
nv50pe8sqr400ID1/pnz20	5	1.02	1.67
nv50pe10sqr400ID1/pnz5	5	1.36	1.79
nv50pe10sqr400ID1/pnz10	5	1.48	1.79
nv50pe10sqr400ID1/pnz15	5	1.52	1.79
nv50pe10sqr400ID1/pnz20	5	1.09	1.79
nv100pe3sqr200ID1/pnz5	5	1.19	1.52
nv100pe3sqr200ID1/pnz10	5	1.19	1.52
nv100pe3sqr200ID1/pnz15	5	1.30	1.52
nv100pe3sqr200ID1/pnz20	5	1.25	1.52
nv100pe5sqr200ID1/pnz5	5	1.36	1.82
nv100pe5sqr200ID1/pnz10	5	1.39	1.82
nv100pe5sqr200ID1/pnz15	5	1.61	1.82
nv100pe5sqr200ID1/pnz20	5	1.45	1.82
nv100pe8sqr200ID1/pnz5	5	1.76	2.22
nv100pe8sqr200ID1/pnz10	5	1.99	2.22
nv100pe8sqr200ID1/pnz15	5	2.15	2.22
nv100pe8sqr200ID1/pnz20	5	1.74	2.22
nv100pe10sqr200ID1/pnz5	5	2.14	2.63
nv100pe10sqr200ID1/pnz10	5	2.50	2.63
nv100pe10sqr200ID1/pnz15	5	2.56	2.63
nv100pe10sqr200ID1/pnz20	5	2.03	2.63
nv125pe3sqr160ID1/pnz5	5	1.34	1.74
nv125pe3sqr160ID1/pnz10	5	1.35	1.74
nv125pe3sqr160ID1/pnz15	5	1.49	1.74
nv125pe3sqr160ID1/pnz20	5	1.52	1.74
nv125pe5sqr160ID1/pnz5	5	1.66	2.16
nv125pe5sqr160ID1/pnz10	5	1.72	2.16
nv125pe5sqr160ID1/pnz15	5	1.99	2.16
nv125pe5sqr160ID1/pnz20	5	1.87	2.16
nv125pe8sqr160ID1/pnz5	5	2.20	2.66
nv125pe8sqr160ID1/pnz10	5	2.51	2.66
nv125pe8sqr160ID1/pnz15	5	2.59	2.66
nv125pe8sqr160ID1/pnz20	5	2.34	2.66
nv125pe10sqr160ID1/pnz5	5	2.29	2.84
nv125pe10sqr160ID1/pnz10	5	2.73	2.84
nv125pe10sqr160ID1/pnz15	5	2.75	2.84
nv125pe10sqr160ID1/pnz20	5	2.46	2.84

Relative performance of various algorithms In Table 8, we show the relative gap of various algorithms. The numbers in brackets record the number of instances where the relative gaps is 0. The geometric mean is computed for instances with non-zero relative gap.

In Table 9, we present the number of times approximation algorithm based strategies beat the other methods.

We can make the following observations:

1. Performance of `CPLEX`: `CPLEX` performs very poorly on these large scale instances. In fact, almost always the only solution it reports is the trivial all zeros solution.

2. Approximation algorithm together with GRASP vs GRASP: Only in 30 instances (out of a total of 240 instances) GRASP produces a better solution than approximation algorithm together with GRASP. In fact, if we see Table 8, we also see that Approximation algorithm together with GRASP produces significantly better solutions that GRASP. It is also interesting to observe that when the relative gap of approximation algorithm is high, it is also usually high for GRASP.

3. Approximation algorithm together with GRASP and local search vs GRASP with local search: Only in 49 instances (out of a total of 240 instances) GRASP with local search produces a better solution than approximation algorithm together with GRASP and local search. In fact, if we see Table 8, we also that Approximation algorithm together with GRASP and local search produces significantly better solutions that GRASP and local search.

4. Effect on performance of approximation algorithm vs number of vertices in $G^{\text{pack}}_{A,\mathcal{J}}$: The solution produced by approximation algorithm is very good when the number of vertices is low. Indeed some instances are solved exactly and

the geometric mean of the relative gap varies between $8\% - 22\%$ for graphs with 50 nodes. For these instances, the improvement produced by GRASP and local search over the solution produced by approximation algorithm, although non-zero, is not substantial. On the other hand, for graphs with 125 nodes, the geometric mean of the relative gap for the approximation algorithm varies between $25\% - 64\%$. We also see that GRASP and local search produces more improvement over approximation algorithm in this case. This trend may be explained by the increase in the fractional mixed chromatic number as the number of vertices increase. In Table 7, we see a indication of this trend, via observing that the lower bounds on the fractional mixed chromatic number increases with increasing number of vertices.

5. Local search is a stronger method to improve the initial solutions than basic GRASP iterations. In particular, if GRASP improves an initial solution considerably, then typically we observe that local search also works effectively.

Time taken by various algorithms In Table 10, we show the time taken by the various algorithms. We make the following observations:

1. One disadvantage of our algorithm is that since it depends on solving sub-IPs, the computational time varies quite a bit.

2. Note that the average time for one iteration of local search is lesser when called after approximation algorithm and GRASP, compared to the when called after basic GRASP. This is because the solution is usually lower in quality after GRASP and therefore local-search runs many more loops.

Table 8: Relative gap of various algorithms (random graph instances)

Instance Features	NIns	Gap^{A}	Gap^{AG}	Gap^{AGS}	Gap^{IP}	Gap^{G}	Gap^{GS}
nv50pe3sqr400ID1/pnz5	5	9.37	9.18	9.10	100.00	19.57	16.92
nv50pe3sqr400ID1/pnz10	5	7.82	7.82	7.81	100.00	19.80	16.74
nv50pe3sqr400ID1/pnz15	5	7.86(1)	7.86(1)	7.86(1)	10.13	25.32	17.07
nv50pe3sqr400ID1/pnz20	5	N/A(5)	N/A(5)	N/A(5)	100.00	1.49	N/A(5)
nv50pe5sqr400ID1/pnz5	5	13.52	12.94	12.32	100.00	21.65	18.58
nv50pe5sqr400ID1/pnz10	5	15.42	14.98	13.99	100.00	23.82	18.45
nv50pe5sqr400ID1/pnz15	5	18.64	18.64	18.64	100.00	32.92	24.72
nv50pe5sqr400ID1/pnz20	5	N/A(5)	N/A(5)	N/A(5)	100.00	8.80	3.70
nv50pe8sqr400ID1/pnz5	5	14.24	13.18	12.68	100.00	21.81	18.37
nv50pe8sqr400ID1/pnz10	5	16.60	16.04	15.45	100.00	23.47	20.12
nv50pe8sqr400ID1/pnz15	5	22.92	22.92	22.92	100.00	36.55	28.74
nv50pe8sqr400ID1/pnz20	5	1.96	1.96	1.96	100.00	6.76	3.04
nv50pe10sqr400ID1/pnz5	5	26.47	25.38	24.75	100.00	30.84	28.60
nv50pe10sqr400ID1/pnz10	5	32.63	32.63	31.72	100.00	37.06	33.18
nv50pe10sqr400ID1/pnz15	5	34.25	34.25	34.25	100.00	43.43	39.26
nv50pe10sqr400ID1/pnz20	5	8.00	8.00	8.00	100.00	14.68	8.00
nv100pe3sqr200ID1/pnz5	5	15.67	14.83	14.07	100.00	22.70	18.92
nv100pe3sqr200ID1/pnz10	5	15.94	15.78	15.49	100.00	26.30	23.16
nv100pe3sqr200ID1/pnz15	5	22.78	22.78	22.14	100.00	30.50	26.02
nv100pe3sqr200ID1/pnz20	5	19.76	19.76	19.76	100.00	33.44	25.94
nv100pe5sqr200ID1/pnz5	5	26.47	23.72	22.82	100.00	29.40	26.26
nv100pe5sqr200ID1/pnz10	5	28.29	27.30	25.06	100.00	31.74	28.50
nv100pe5sqr200ID1/pnz15	5	37.70	37.68	36.96	100.00	42.61	38.34
nv100pe5sqr200ID1/pnz20	5	30.81	30.81	30.81	100.00	38.13	34.13
nv100pe8sqr200ID1/pnz5	5	43.27	35.10	33.45	100.00	37.97	35.23
nv100pe8sqr200ID1/pnz10	5	49.76	45.97	41.66	100.00	43.67	40.19
nv100pe8sqr200ID1/pnz15	5	53.38	53.38	52.57	100.00	55.17	52.08
nv100pe8sqr200ID1/pnz20	5	42.60	42.60	42.60	100.00	51.10	47.48
nv100pe10sqr200ID1/pnz5	5	53.18	41.52	38.91	100.00	41.63	39.12
nv100pe10sqr200ID1/pnz10	5	60.01	56.45	52.25	100.00	51.42	48.93
nv100pe10sqr200ID1/pnz15	5	60.89	60.89	60.78	100.00	63.05	61.54
nv100pe10sqr200ID1/pnz20	5	50.66	50.66	50.66	100.00	57.34	54.91
nv125pe3sqr160ID1/pnz5	5	25.56	23.04	21.63	100.00	27.42	24.64
nv125pe3sqr160ID1/pnz10	5	25.83	24.79	23.67	100.00	30.54	26.95
nv125pe3sqr160ID1/pnz15	5	32.80	32.31	31.52	100.00	37.71	33.26
nv125pe3sqr160ID1/pnz20	5	34.04	34.04	33.99	100.00	43.62	40.33
nv125pe5sqr160ID1/pnz5	5	39.74	33.12	31.46	100.00	35.25	32.47
nv125pe5sqr160ID1/pnz10	5	41.87	38.16	35.64	100.00	38.66	35.33
nv125pe5sqr160ID1/pnz15	5	49.63	49.22	47.85	100.00	49.91	47.09
nv125pe5sqr160ID1/pnz20	5	46.64	46.64	46.57	100.00	53.44	51.20
nv125pe8sqr160ID1/pnz5	5	54.46	42.16	40.43	100.00	42.13	39.57
nv125pe8sqr160ID1/pnz10	5	60.16	53.57	49.18	100.00	48.97	46.72
nv125pe8sqr160ID1/pnz15	5	61.40	61.23	60.11	100.00	62.07	59.74
nv125pe8sqr160ID1/pnz20	5	57.34	57.34	57.20	100.00	63.83	61.24
nv125pe10sqr160ID1/pnz5	5	56.35	43.19	40.07	100.00	42.83	40.46
nv125pe10sqr160ID1/pnz10	5	63.40	58.30	53.00	100.00	51.14	48.38
nv125pe10sqr160ID1/pnz15	5	63.66	63.66	62.39	100.00	63.72	61.34
nv125pe10sqr160ID1/pnz20	5	59.39	59.39	59.39	100.00	64.79	62.55

Table 9: Number of instances where approximation algorithm based strategy is better than CPLEX or pure GRASP based strategy (random graph instances)

Instance Features	NIns	N(AG $\geq$ IP)	M(AG $\geq$ G)	N(AGS $\geq$ GS)
nv50pe3sqr400ID1/pnz5	5	5	5	5
nv50pe3sqr400ID1/pnz10	5	5	5	5
nv50pe3sqr400ID1/pnz15	5	5	5	5
nv50pe3sqr400ID1/pnz20	5	5	5	5
nv50pe5sqr400ID1/pnz5	5	5	5	5
nv50pe5sqr400ID1/pnz10	5	5	5	5
nv50pe5sqr400ID1/pnz15	5	5	5	5
nv50pe5sqr400ID1/pnz20	5	5	5	5
nv50pe8sqr400ID1/pnz5	5	5	5	5
nv50pe8sqr400ID1/pnz10	5	5	5	5
nv50pe8sqr400ID1/pnz15	5	5	5	5
nv50pe8sqr400ID1/pnz20	5	5	5	5
nv50pe10sqr400ID1/pnz5	5	5	5	5
nv50pe10sqr400ID1/pnz10	5	5	5	5
nv50pe10sqr400ID1/pnz15	5	5	5	5
nv50pe10sqr400ID1/pnz20	5	5	5	5
nv100pe3sqr200ID1/pnz5	5	5	5	5
nv100pe3sqr200ID1/pnz10	5	5	5	5
nv100pe3sqr200ID1/pnz15	5	5	5	5
nv100pe3sqr200ID1/pnz20	5	5	5	5
nv100pe5sqr200ID1/pnz5	5	5	5	5
nv100pe5sqr200ID1/pnz10	5	5	5	5
nv100pe5sqr200ID1/pnz15	5	5	5	4
nv100pe5sqr200ID1/pnz20	5	5	5	5
nv100pe8sqr200ID1/pnz5	5	5	5	5
nv100pe8sqr200ID1/pnz10	5	5	1	1
nv100pe8sqr200ID1/pnz15	5	5	5	1
nv100pe8sqr200ID1/pnz20	5	5	5	5
nv100pe10sqr200ID1/pnz5	5	5	4	3
nv100pe10sqr200ID1/pnz10	5	5	0	0
nv100pe10sqr200ID1/pnz15	5	5	5	5
nv100pe10sqr200ID1/pnz20	5	5	5	5
nv125pe3sqr160ID1/pnz5	5	5	5	5
nv125pe3sqr160ID1/pnz10	5	5	5	5
nv125pe3sqr160ID1/pnz15	5	5	5	4
nv125pe3sqr160ID1/pnz20	5	5	5	5
nv125pe5sqr160ID1/pnz5	5	5	5	5
nv125pe5sqr160ID1/pnz10	5	5	3	2
nv125pe5sqr160ID1/pnz15	5	5	5	0
nv125pe5sqr160ID1/pnz20	5	5	5	5
nv125pe8sqr160ID1/pnz5	5	5	2	1
nv125pe8sqr160ID1/pnz10	5	5	0	0
nv125pe8sqr160ID1/pnz15	5	5	5	2
nv125pe8sqr160ID1/pnz20	5	5	5	5
nv125pe10sqr160ID1/pnz5	5	5	2	3
nv125pe10sqr160ID1/pnz10	5	5	0	0
nv125pe10sqr160ID1/pnz15	5	5	3	0
nv125pe10sqr160ID1/pnz20	5	5	5	5

Table 10: Computational time (random graph instances)

Instance Features	NIns	T^{A}_{AGS}	T^{G}_{AGS}	T^{S}_{AGS}	$T^{G}_{G}(NIt^{G}_{G})$	$T^{G}_{GS}(NIt^{G}_{GS})$	$T^{S}_{GS}(NIt^{S}_{GS})$
nv50pe3sqr400ID1/pnz5	5	20457.85	3.50	182.12	21484.75(50648)	17271.19(47387)	4317.6(6)
nv50pe3sqr400ID1/pnz10	5	615.50	6.10	44.84	653.84(4730)	262.03(1906)	534.03(2)
nv50pe3sqr400ID1/pnz15	5	297.71	9.17	30.26	323.64(2378)	76.39(686)	545.24(2)
nv50pe3sqr400ID1/pnz20	5	309.31	11.90	19.99	339.69(2527)	41.43(343)	357.15(1)
nv50pe5sqr400ID1/pnz5	5	16818.35	3.71	291.24	17663.5(50236)	15697.24(44556)	2339.53(4)
nv50pe5sqr400ID1/pnz10	5	688.84	6.76	62.99	731.68(4908)	244.12(1805)	562.3(3)
nv50pe5sqr400ID1/pnz15	5	334.25	9.90	27.10	363.16(2962)	78.22(708)	370.98(2)
nv50pe5sqr400ID1/pnz20	5	383.46	13.31	18.52	419.52(3082)	41.4(343)	474.46(1)
nv50pe8sqr400ID1/pnz5	5	10451.51	4.61	213.37	10979.62(28869)	7875.48(22827)	3406.78(5)
nv50pe8sqr400ID1/pnz10	5	956.87	8.67	61.04	1015.81(6410)	555.49(3937)	561.91(3)
nv50pe8sqr400ID1/pnz15	5	503.22	12.71	25.06	544.24(4017)	182.57(1524)	476.42(3)
nv50pe8sqr400ID1/pnz20	5	617.50	17.08	18.47	669.88(4985)	149.4(1112)	553.63(2)
nv50pe10sqr400ID1/pnz5	5	4937.90	5.60	109.82	5191.53(27183)	4024.98(21182)	1261.2(5)
nv50pe10sqr400ID1/pnz10	5	1190.16	10.55	44.70	1263.15(11798)	671.41(6291)	551.36(3)
nv50pe10sqr400ID1/pnz15	5	704.49	15.83	19.89	759.76(6447)	421.45(3590)	288.11(1)
nv50pe10sqr400ID1/pnz20	5	872.72	21.34	17.02	943.27(6871)	192.33(1401)	991.01(2)
nv100pe3sqr200ID1/pnz5	5	10332.09	2.34	2346.63	10851.59(10167)	1036.58(648)	17734.07(2)
nv100pe3sqr200ID1/pnz10	5	550.77	3.90	221.97	582.8(1605)	105.5(324)	1012.13(1)
nv100pe3sqr200ID1/pnz15	5	634.25	5.39	126.20	672.45(3317)	107.68(532)	1165.15(2)
nv100pe3sqr200ID1/pnz20	5	463.56	7.23	53.83	495.52(2667)	43.37(260)	923.27(1)
nv100pe5sqr200ID1/pnz5	5	10535.67	4.90	1306.72	11068.18(12638)	3522.41(4850)	9085.36(3)
nv100pe5sqr200ID1/pnz10	5	882.22	8.24	460.31	936.26(3805)	214.87(612)	1667(2)
nv100pe5sqr200ID1/pnz15	5	1685.53	12.62	157.06	1785.05(10826)	196.93(1074)	2250.24(3)
nv100pe5sqr200ID1/pnz20	5	1326.51	16.70	69.26	1412.78(9043)	66.73(260)	2543.14(1)
nv100pe8sqr200ID1/pnz5	5	3553.59	5.50	617.17	3737.95(13156)	2162.88(7618)	2515.7(3)
nv100pe8sqr200ID1/pnz10	5	1185.97	8.86	304.34	1256.55(9589)	258.67(1970)	1413.52(3)
nv100pe8sqr200ID1/pnz15	5	2343.87	12.58	65.54	2477.04(19449)	622.41(4858)	2144.48(5)
nv100pe8sqr200ID1/pnz20	5	2135.08	17.18	32.57	2263.72(16065)	512.15(3627)	1767.4(3)
nv100pe10sqr200ID1/pnz5	5	3207.00	6.84	586.40	3375.61(10797)	1906.49(8345)	2148.72(4)
nv100pe10sqr200ID1/pnz10	5	2041.85	10.85	166.05	2157.58(17703)	810.83(6651)	1491.27(4)
nv100pe10sqr200ID1/pnz15	5	3915.26	15.92	30.71	4131.28(33100)	2818.83(22561)	1114.05(3)
nv100pe10sqr200ID1/pnz20	5	3470.92	21.39	26.91	3671.68(25589)	2003.39(13937)	1656.48(4)
nv125pe3sqr160ID1/pnz5	5	5439.59	3.10	2946.31	5715.69(4628)	821.85(724)	9117.28(2)
nv125pe3sqr160ID1/pnz10	5	577.54	4.19	369.21	611.67(1408)	92.01(260)	1262.88(1)
nv125pe3sqr160ID1/pnz15	5	856.73	5.90	233.93	906.72(4217)	114.85(532)	1635.46(2)
nv125pe3sqr160ID1/pnz20	5	1102.95	7.83	57.14	1167.41(6384)	54.61(299)	2095.39(1)
nv125pe5sqr160ID1/pnz5	5	4069.71	4.65	1296.16	4278.82(6197)	982.47(1304)	5815.47(2)
nv125pe5sqr160ID1/pnz10	5	941.76	6.55	423.25	996.92(3521)	328.74(1414)	1292.34(3)
nv125pe5sqr160ID1/pnz15	5	1851.34	9.15	194.69	1955.46(11943)	310.79(1892)	1978.64(3)
nv125pe5sqr160ID1/pnz20	5	2342.56	12.24	48.49	2475.09(15808)	275.09(1387)	2440.62(2)
nv125pe8sqr160ID1/pnz5	5	4376.76	8.06	863.72	4605.28(17906)	1611.34(5630)	3669.14(3)
nv125pe8sqr160ID1/pnz10	5	3696.67	11.63	440.89	3896.34(29753)	1624.12(9984)	2027.42(4)
nv125pe8sqr160ID1/pnz15	5	7897.22	17.77	118.54	8315.12(52750)	5198.99(40655)	2946.79(5)
nv125pe8sqr160ID1/pnz20	5	7595.96	20.41	49.37	8001.76(56738)	4510.29(28305)	3144.46(4)
nv125pe10sqr160ID1/pnz5	5	4055.80	7.71	993.22	4267.97(15699)	1242.16(4505)	4099.93(4)
nv125pe10sqr160ID1/pnz10	5	7006.58	11.39	314.94	7371.43(57298)	5289.29(41137)	2063.22(5)
nv125pe10sqr160ID1/pnz15	5	11612.76	16.23	92.06	12214.37(87387)	9792.94(76028)	2308.64(5)
nv125pe10sqr160ID1/pnz20	5	12945.30	21.94	36.31	13621.52(84984)	9319.72(56751)	3716.58(5)

3.4.5.3 *Results for three-layer tree instances*

Approximate algorithm's actual performance vs theoretical worst case performance In Table 11 and Table 12, we compare the value of $\frac{BestBound}{z^A}$ against the fractional mixed chromatic number. The values of $\frac{BestBound}{z^A}$ are lower than the fractional mixed chromatic number with respect to $\mathcal{V}$, indicting that the actual performance of the algorithm is significantly better than that of the worst case performance guarantee.

Table 11: $\frac{BestBound}{z^A}$ vs fractional mixed chromatic number I (tree instances)

Instance Features	NIns	M(B/A)	η(G)
ss8ts6nv57sqr100/pnz5	5	1.08	1.88
ss8ts6nv57sqr100/pnz10	5	1.13	1.88
ss8ts6nv57sqr100/pnz15	5	1.15	1.88
ss8ts6nv57sqr100/pnz20	5	1.22	1.88
ss8ts6nv57sqr150/pnz5	5	1.12	1.88
ss8ts6nv57sqr150/pnz10	5	1.12	1.88
ss8ts6nv57sqr150/pnz15	5	1.19	1.88
ss8ts6nv57sqr150/pnz20	5	1.27	1.88
ss8ts6nv57sqr200/pnz5	5	1.12	1.88
ss8ts6nv57sqr200/pnz10	5	1.13	1.88
ss8ts6nv57sqr200/pnz15	5	1.21	1.88
ss8ts6nv57sqr200/pnz20	5	1.28	1.88
ss8ts8nv73sqr100/pnz5	5	1.07	1.89
ss8ts8nv73sqr100/pnz10	5	1.10	1.89
ss8ts8nv73sqr100/pnz15	5	1.16	1.89
ss8ts8nv73sqr100/pnz20	5	1.23	1.89
ss8ts8nv73sqr150/pnz5	5	1.11	1.89
ss8ts8nv73sqr150/pnz10	5	1.11	1.89
ss8ts8nv73sqr150/pnz15	5	1.19	1.89
ss8ts8nv73sqr150/pnz20	5	1.28	1.89
ss8ts8nv73sqr200/pnz5	5	1.10	1.89
ss8ts8nv73sqr200/pnz10	5	1.13	1.89
ss8ts8nv73sqr200/pnz15	5	1.22	1.89
ss8ts8nv73sqr200/pnz20	5	1.29	1.89
ss8ts10nv89sqr100/pnz5	5	1.08	1.91
ss8ts10nv89sqr100/pnz10	5	1.10	1.91
ss8ts10nv89sqr100/pnz15	5	1.16	1.91
ss8ts10nv89sqr100/pnz20	5	1.23	1.91
ss8ts10nv89sqr150/pnz5	5	1.09	1.91
ss8ts10nv89sqr150/pnz10	5	1.12	1.91
ss8ts10nv89sqr150/pnz15	5	1.20	1.91
ss8ts10nv89sqr150/pnz20	5	1.29	1.91
ss8ts10nv89sqr200/pnz5	5	1.09	1.91
ss8ts10nv89sqr200/pnz10	5	1.14	1.91
ss8ts10nv89sqr200/pnz15	5	1.23	1.91
ss8ts10nv89sqr200/pnz20	5	1.30	1.91

Table 12: $\frac{BestBound}{z^A}$ vs fractional mixed chromatic number II (tree instances)

Instance Features	NIns	M(B/A)	η(G)
ss10ts6nv71sqr100/pnz5	5	1.07	1.90
ss10ts6nv71sqr100/pnz10	5	1.13	1.90
ss10ts6nv71sqr100/pnz15	5	1.16	1.90
ss10ts6nv71sqr100/pnz20	5	1.21	1.90
ss10ts6nv71sqr150/pnz5	5	1.13	1.90
ss10ts6nv71sqr150/pnz10	5	1.12	1.90
ss10ts6nv71sqr150/pnz15	5	1.18	1.90
ss10ts6nv71sqr150/pnz20	5	1.27	1.90
ss10ts6nv71sqr200/pnz5	5	1.12	1.90
ss10ts6nv71sqr200/pnz10	5	1.12	1.90
ss10ts6nv71sqr200/pnz15	5	1.21	1.90
ss10ts6nv71sqr200/pnz20	5	1.28	1.90
ss10ts8nv91sqr100/pnz5	5	1.07	1.90
ss10ts8nv91sqr100/pnz10	5	1.10	1.90
ss10ts8nv91sqr100/pnz15	5	1.16	1.90
ss10ts8nv91sqr100/pnz20	5	1.23	1.90
ss10ts8nv91sqr150/pnz5	5	1.11	1.90
ss10ts8nv91sqr150/pnz10	5	1.11	1.90
ss10ts8nv91sqr150/pnz15	5	1.19	1.90
ss10ts8nv91sqr150/pnz20	5	1.28	1.90
ss10ts8nv91sqr200/pnz5	5	1.11	1.90
ss10ts8nv91sqr200/pnz10	5	1.13	1.90
ss10ts8nv91sqr200/pnz15	5	1.22	1.90
ss10ts8nv91sqr200/pnz20	5	1.29	1.90
ss10ts10nv111sqr100/pnz5	5	1.05	1.91
ss10ts10nv111sqr100/pnz10	5	1.10	1.91
ss10ts10nv111sqr100/pnz15	5	1.17	1.91
ss10ts10nv111sqr100/pnz20	5	1.23	1.91
ss10ts10nv111sqr150/pnz5	5	1.10	1.91
ss10ts10nv111sqr150/pnz10	5	1.12	1.91
ss10ts10nv111sqr150/pnz15	5	1.20	1.91
ss10ts10nv111sqr150/pnz20	5	1.28	1.91
ss10ts10nv111sqr200/pnz5	5	1.09	1.91
ss10ts10nv111sqr200/pnz10	5	1.14	1.91
ss10ts10nv111sqr200/pnz15	5	1.22	1.91
ss10ts10nv111sqr200/pnz20	5	1.30	1.91

Relative performance of various algorithms In Table 13 and Table 14, we show the relative gap of various algorithms.

Table 13: Relative gap of various algorithms I (tree instances)

Instance Features	NIns	$\mathtt{Gap}^{\mathtt{A}}$	$\mathtt{Gap}^{\mathtt{AG}}$	$\mathtt{Gap}^{\mathtt{AGS}}$	$\mathtt{Gap}^{\mathtt{IP}}$	$\mathtt{Gap}^{\mathtt{G}}$	$\mathtt{Gap}^{\mathtt{GS}}$
ss8ts6nv57sqr100/pnz5	5	7.44	7.44	7.44	11.59	15.53	10.08
ss8ts6nv57sqr100/pnz10	5	11.21	11.21	11.21	99.93	22.24	16.62
ss8ts6nv57sqr100/pnz15	5	13.39	13.39	13.39	100.00	20.48	17.37
ss8ts6nv57sqr100/pnz20	5	18.00	18.00	18.00	64.85	26.18	22.26
ss8ts6nv57sqr150/pnz5	5	10.95	10.95	10.95	21.13	20.18	16.24
ss8ts6nv57sqr150/pnz10	5	10.85	10.85	10.85	80.16	21.29	17.02
ss8ts6nv57sqr150/pnz15	5	15.66	15.66	15.66	100.00	21.90	19.05
ss8ts6nv57sqr150/pnz20	5	21.12	21.12	21.12	100.00	22.09	21.34
ss8ts6nv57sqr200/pnz5	5	10.47	10.47	10.47	30.31	20.74	16.28
ss8ts6nv57sqr200/pnz10	5	11.23	11.23	11.23	100.00	21.43	17.27
ss8ts6nv57sqr200/pnz15	5	17.32	17.32	17.32	100.00	28.80	23.12
ss8ts6nv57sqr200/pnz20	5	21.74	21.74	21.74	61.26	31.82	21.74
ss8ts8nv73sqr100/pnz5	5	6.81	6.81	6.81	9.02	14.44	9.65
ss8ts8nv73sqr100/pnz10	5	9.13	9.13	9.13	100.00	19.80	13.94
ss8ts8nv73sqr100/pnz15	5	13.82	13.82	13.82	59.00	21.72	16.78
ss8ts8nv73sqr100/pnz20	5	18.57	18.57	18.57	50.29	26.89	22.73
ss8ts8nv73sqr150/pnz5	5	9.70	9.70	9.70	25.76	19.57	14.60
ss8ts8nv73sqr150/pnz10	5	9.78	9.78	9.78	100.00	19.69	15.24
ss8ts8nv73sqr150/pnz15	5	16.29	16.29	16.29	100.00	22.50	18.96
ss8ts8nv73sqr150/pnz20	5	21.96	21.96	21.96	100.00	23.01	22.36
ss8ts8nv73sqr200/pnz5	5	9.37	9.37	9.37	61.49	20.36	15.43
ss8ts8nv73sqr200/pnz10	5	11.84	11.84	11.84	100.00	22.24	18.03
ss8ts8nv73sqr200/pnz15	5	17.89	17.89	17.89	100.00	29.07	24.07
ss8ts8nv73sqr200/pnz20	5	22.65	22.65	22.65	100.00	33.49	22.65
ss8ts10nv89sqr100/pnz5	5	7.67	7.67	7.67	11.24	15.67	11.12
ss8ts10nv89sqr100/pnz10	5	8.94	8.94	8.94	53.47	19.76	13.89
ss8ts10nv89sqr100/pnz15	5	14.06	14.06	14.06	61.92	22.01	18.27
ss8ts10nv89sqr100/pnz20	5	18.96	18.96	18.96	51.93	27.81	23.74
ss8ts10nv89sqr150/pnz5	5	8.65	8.65	8.65	26.78	19.77	13.27
ss8ts10nv89sqr150/pnz10	5	10.42	10.42	10.42	99.97	20.63	16.53
ss8ts10nv89sqr150/pnz15	5	16.50	16.50	16.50	100.00	23.98	20.04
ss8ts10nv89sqr150/pnz20	5	22.55	22.55	22.55	100.00	24.09	22.81
ss8ts10nv89sqr200/pnz5	5	8.14	8.14	8.14	100.00	19.26	14.43
ss8ts10nv89sqr200/pnz10	5	12.02	12.02	12.02	100.00	22.61	18.36
ss8ts10nv89sqr200/pnz15	5	18.57	18.57	18.57	100.00	30.58	24.95
ss8ts10nv89sqr200/pnz20	5	23.12	23.12	23.12	83.86	32.82	23.12

Table 14: Relative gap of various algorithms II (tree instances)

Instance Features	NIns	Gap^{A}	Gap^{AG}	Gap^{AGS}	Gap^{IP}	Gap^{G}	Gap^{GS}
ss10ts6nv71sqr100/pnz5	5	6.38	6.38	6.38	10.30	13.43	9.55
ss10ts6nv71sqr100/pnz10	5	11.47	11.47	11.47	100.00	22.50	16.83
ss10ts6nv71sqr100/pnz15	5	13.42	13.42	13.42	67.74	21.23	17.21
ss10ts6nv71sqr100/pnz20	5	17.63	17.63	17.63	58.49	26.68	22.57
ss10ts6nv71sqr150/pnz5	5	11.29	11.29	11.29	24.17	20.34	15.83
ss10ts6nv71sqr150/pnz10	5	11.08	11.08	11.08	100.00	20.86	17.38
ss10ts6nv71sqr150/pnz15	5	15.50	15.50	15.50	100.00	20.81	18.10
ss10ts6nv71sqr150/pnz20	5	20.96	20.96	20.96	100.00	22.44	21.40
ss10ts6nv71sqr200/pnz5	5	10.31	10.31	10.31	42.88	22.41	16.61
ss10ts6nv71sqr200/pnz10	5	11.07	11.07	11.07	100.00	22.48	18.81
ss10ts6nv71sqr200/pnz15	5	17.42	17.42	17.42	100.00	28.69	22.79
ss10ts6nv71sqr200/pnz20	5	21.76	21.76	21.76	50.44	32.53	21.76
ss10ts8nv91sqr100/pnz5	5	6.56	6.56	6.56	8.30	13.58	9.71
ss10ts8nv91sqr100/pnz10	5	9.14	9.14	9.14	61.35	20.67	14.42
ss10ts8nv91sqr100/pnz15	5	13.96	13.96	13.96	69.74	22.69	18.11
ss10ts8nv91sqr100/pnz20	5	18.42	18.42	18.42	46.14	27.33	23.83
ss10ts8nv91sqr150/pnz5	5	9.99	9.99	9.99	32.27	19.45	14.55
ss10ts8nv91sqr150/pnz10	5	10.15	10.15	10.15	100.00	20.47	16.38
ss10ts8nv91sqr150/pnz15	5	16.24	16.24	16.24	100.00	22.69	19.39
ss10ts8nv91sqr150/pnz20	5	21.63	21.63	21.63	78.77	23.30	22.02
ss10ts8nv91sqr200/pnz5	5	9.73	9.73	9.73	78.43	21.54	16.65
ss10ts8nv91sqr200/pnz10	5	11.69	11.69	11.69	100.00	22.46	17.85
ss10ts8nv91sqr200/pnz15	5	18.02	18.02	18.02	100.00	30.38	24.22
ss10ts8nv91sqr200/pnz20	5	22.60	22.60	22.60	84.23	33.53	22.60
ss10ts10nv111sqr100/pnz5	5	4.59	4.59	4.59	7.48	15.19	7.46
ss10ts10nv111sqr100/pnz10	5	9.37	9.37	9.37	81.69	18.97	13.94
ss10ts10nv111sqr100/pnz15	5	14.21	14.21	14.21	61.34	22.38	18.29
ss10ts10nv111sqr100/pnz20	5	18.93	18.93	18.93	46.92	29.00	23.97
ss10ts10nv111sqr150/pnz5	5	9.13	9.13	9.13	45.19	18.39	13.77
ss10ts10nv111sqr150/pnz10	5	10.48	10.48	10.48	100.00	20.93	17.01
ss10ts10nv111sqr150/pnz15	5	16.77	16.77	16.77	100.00	24.28	20.15
ss10ts10nv111sqr150/pnz20	5	22.16	22.16	22.16	58.17	24.20	22.67
ss10ts10nv111sqr200/pnz5	5	8.44	8.44	8.44	100.00	21.15	14.97
ss10ts10nv111sqr200/pnz10	5	12.04	12.04	12.04	100.00	22.44	18.40
ss10ts10nv111sqr200/pnz15	5	18.33	18.33	18.33	100.00	30.97	24.57
ss10ts10nv111sqr200/pnz20	5	23.11	23.11	23.11	71.86	34.29	23.11

In Table 15 and Table 16, we present the number of times approximation algorithm based strategies beat the other methods.

Table 15: Number of instances where approximation algorithm based strategy is better than CPLEX or pure GRASP based strategy I (tree instances)

Instance Features	NIns	N(AG $\geq$ IP)	M(AG $\geq$ G)	N(AGS $\geq$ GS)
ss8ts6nv57sqr100/pnz5	5	5	5	5
ss8ts6nv57sqr100/pnz10	5	5	5	5
ss8ts6nv57sqr100/pnz15	5	5	5	5
ss8ts6nv57sqr100/pnz20	5	5	5	5
ss8ts6nv57sqr150/pnz5	5	5	5	5
ss8ts6nv57sqr150/pnz10	5	5	5	5
ss8ts6nv57sqr150/pnz15	5	5	5	5
ss8ts6nv57sqr150/pnz20	5	5	5	5
ss8ts6nv57sqr200/pnz5	5	5	5	5
ss8ts6nv57sqr200/pnz10	5	5	5	5
ss8ts6nv57sqr200/pnz15	5	5	5	5
ss8ts6nv57sqr200/pnz20	5	5	5	5
ss8ts8nv73sqr100/pnz5	5	5	5	5
ss8ts8nv73sqr100/pnz10	5	5	5	5
ss8ts8nv73sqr100/pnz15	5	5	5	5
ss8ts8nv73sqr100/pnz20	5	5	5	5
ss8ts8nv73sqr150/pnz5	5	5	5	5
ss8ts8nv73sqr150/pnz10	5	5	5	5
ss8ts8nv73sqr150/pnz15	5	5	5	5
ss8ts8nv73sqr150/pnz20	5	5	5	5
ss8ts8nv73sqr200/pnz5	5	5	5	5
ss8ts8nv73sqr200/pnz10	5	5	5	5
ss8ts8nv73sqr200/pnz15	5	5	5	5
ss8ts8nv73sqr200/pnz20	5	5	5	5
ss8ts10nv89sqr100/pnz5	5	5	5	5
ss8ts10nv89sqr100/pnz10	5	5	5	5
ss8ts10nv89sqr100/pnz15	5	5	5	5
ss8ts10nv89sqr100/pnz20	5	5	5	5
ss8ts10nv89sqr150/pnz5	5	5	5	5
ss8ts10nv89sqr150/pnz10	5	5	5	5
ss8ts10nv89sqr150/pnz15	5	5	5	5
ss8ts10nv89sqr150/pnz20	5	5	5	5
ss8ts10nv89sqr200/pnz5	5	5	5	5
ss8ts10nv89sqr200/pnz10	5	5	5	5
ss8ts10nv89sqr200/pnz15	5	5	5	5
ss8ts10nv89sqr200/pnz20	5	5	5	5

Table 16: Number of instances where approximation algorithm based strategy is better than CPLEX or pure GRASP based strategy II (tree instances)

Instance Features	NIns	N(AG $\geq$ IP)	M(AG $\geq$ G)	N(AGS $\geq$ GS)
ss10ts6nv71sqr100/pnz5	5	5	5	5
ss10ts6nv71sqr100/pnz10	5	5	5	5
ss10ts6nv71sqr100/pnz15	5	5	5	5
ss10ts6nv71sqr100/pnz20	5	5	5	5
ss10ts6nv71sqr150/pnz5	5	5	5	5
ss10ts6nv71sqr150/pnz10	5	5	5	5
ss10ts6nv71sqr150/pnz15	5	5	5	5
ss10ts6nv71sqr150/pnz20	5	5	5	5
ss10ts6nv71sqr200/pnz5	5	5	5	5
ss10ts6nv71sqr200/pnz10	5	5	5	5
ss10ts6nv71sqr200/pnz15	5	5	5	5
ss10ts6nv71sqr200/pnz20	5	5	5	5
ss10ts8nv91sqr100/pnz5	5	5	5	5
ss10ts8nv91sqr100/pnz10	5	5	5	5
ss10ts8nv91sqr100/pnz15	5	5	5	5
ss10ts8nv91sqr100/pnz20	5	5	5	5
ss10ts8nv91sqr150/pnz5	5	5	5	5
ss10ts8nv91sqr150/pnz10	5	5	5	5
ss10ts8nv91sqr150/pnz15	5	5	5	5
ss10ts8nv91sqr150/pnz20	5	5	5	5
ss10ts8nv91sqr200/pnz5	5	5	5	5
ss10ts8nv91sqr200/pnz10	5	5	5	5
ss10ts8nv91sqr200/pnz15	5	5	5	5
ss10ts8nv91sqr200/pnz20	5	5	5	5
ss10ts10nv111sqr100/pnz5	5	5	5	5
ss10ts10nv111sqr100/pnz10	5	5	5	5
ss10ts10nv111sqr100/pnz15	5	5	5	5
ss10ts10nv111sqr100/pnz20	5	5	5	5
ss10ts10nv111sqr150/pnz5	5	5	5	5
ss10ts10nv111sqr150/pnz10	5	5	5	5
ss10ts10nv111sqr150/pnz15	5	5	5	5
ss10ts10nv111sqr150/pnz20	5	5	5	5
ss10ts10nv111sqr200/pnz5	5	5	5	5
ss10ts10nv111sqr200/pnz10	5	5	5	5
ss10ts10nv111sqr200/pnz15	5	5	5	5
ss10ts10nv111sqr200/pnz20	5	5	5	5

We can make the following observations:

1. Performance of `CPLEX`: `CPLEX` performs a bit better compared to its performance for random graph instances. However, even then it never produces a better solution that the approximation algorithm.

2. Approximation algorithm vs GRASP: GRASP never produces a better solution than approximation algorithm. We see that approximation algorithm consistently provides a better solution, of the order of 10% to 15% better than GRASP.

3. Approximation algorithm vs GRASP with local search: GRASP with local search never produces a better solution than approximation algorithm.

4. Effect of GRASP and local search on improving the solution produced by approximation algorithm: GRASP and its variants never improve he solution produced by approximation algorithm.

5. Approximation algorithm performs quite well: The geometric mean of relative gap for approximation algorithm is in the range of 6%−23%. This is significantly better performance in comparison to the performance of the approximation algorithm for the random graph instances.

6. Other trends: As long as the size of the block is fixed, the relative gap of approximation algorithm is similar on all different graphs. As long as the graph is fixed, the relative gap of approximation algorithm increases with increasing block size.

Time taken by various algorithms In Table 17 and Table 18, we show the time taken by the various algorithms. We can see that the actual running for GRASP and local search may be as much as 5000% of approximation algorithm, while still providing a poorer solution. Furthermore, this time is consumed for a single iteration of local search. Thus for these instances, local search does not seem to do very well.

Table 17: Computational time I (tree instances)

Instance Features	NIns	T^{A}_{AGS}	T^{G}_{AGS}	T^{S}_{AGS}	$T^{G}_{G}(NIt^{G}_{G})$	$T^{G}_{GS}(NIt^{G}_{GS})$	$T^{S}_{GS}(NIt^{S}_{GS})$
ss8ts6nv57sqr100/pnz5	5	139.05	0.32	280.93	147.29(67)	574.07(260)	9447.6(1)
ss8ts6nv57sqr100/pnz10	5	22.05	0.46	58.73	23.48(56)	108.34(260)	923.1(1)
ss8ts6nv57sqr100/pnz15	5	30.16	0.62	22.21	32.08(192)	43.17(260)	152.35(1)
ss8ts6nv57sqr100/pnz20	5	37.85	0.79	12.29	40.48(399)	26.48(260)	74.23(1)
ss8ts6nv57sqr150/pnz5	5	476.22	0.65	430.03	501.6(177)	673.03(260)	11811.13(1)
ss8ts6nv57sqr150/pnz10	5	143.70	0.94	77.16	152.07(322)	122.94(260)	829.07(1)
ss8ts6nv57sqr150/pnz15	5	82.88	1.28	29.96	88.24(416)	67.71(260)	175.88(1)
ss8ts6nv57sqr150/pnz20	5	112.04	1.65	16.21	119.48(683)	124.31(780)	30.27(1)
ss8ts6nv57sqr200/pnz5	5	875.77	1.03	518.72	922.44(260)	750.37(260)	13905.13(1)
ss8ts6nv57sqr200/pnz10	5	480.21	1.58	94.55	505.7(951)	139.03(260)	960.81(1)
ss8ts6nv57sqr200/pnz15	5	176.95	2.20	43.14	188.36(600)	87.63(260)	414.28(1)
ss8ts6nv57sqr200/pnz20	5	110.90	2.91	28.05	119.63(518)	67.05(260)	371.17(1)
ss8ts8nv73sqr100/pnz5	5	153.75	0.39	607.29	165.87(30)	1465.96(260)	32885.24(1)
ss8ts8nv73sqr100/pnz10	5	36.56	0.60	135.03	38.77(42)	242.06(260)	2697.05(1)
ss8ts8nv73sqr100/pnz15	5	45.26	0.79	49.90	48.26(131)	112.33(260)	728.86(1)
ss8ts8nv73sqr100/pnz20	5	55.60	1.01	27.58	59.24(266)	64.3(260)	241.67(1)
ss8ts8nv73sqr150/pnz5	5	467.33	0.82	940.33	493.89(75)	1919.12(260)	47358.96(1)
ss8ts8nv73sqr150/pnz10	5	232.57	1.23	178.97	245.77(211)	301.75(260)	2922.24(1)
ss8ts8nv73sqr150/pnz15	5	119.92	1.69	65.79	127.78(269)	151.04(260)	619.34(1)
ss8ts8nv73sqr150/pnz20	5	164.22	2.15	36.02	174.82(451)	175.25(453)	97.6(1)
ss8ts8nv73sqr200/pnz5	5	815.59	1.23	1109.47	862.03(128)	1739.96(260)	45197.58(1)
ss8ts8nv73sqr200/pnz10	5	653.90	2.04	211.64	689.39(525)	347.74(260)	2850.19(1)
ss8ts8nv73sqr200/pnz15	5	255.60	2.85	97.05	272.13(386)	195.43(260)	1070.95(1)
ss8ts8nv73sqr200/pnz20	5	172.25	3.80	61.76	185.4(367)	131.79(260)	945.71(1)
ss8ts10nv89sqr100/pnz5	5	159.87	0.48	1149.68	170.5(16)	2750.39(260)	68158.77(1)
ss8ts10nv89sqr100/pnz10	5	61.10	0.79	271.14	65.8(34)	544.38(260)	7015.16(1)
ss8ts10nv89sqr100/pnz15	5	63.46	1.00	95.41	67.42(100)	173.96(260)	1099.69(1)
ss8ts10nv89sqr100/pnz20	5	86.56	1.42	57.86	92.42(224)	119.08(260)	532.65(1)
ss8ts10nv89sqr150/pnz5	5	345.82	0.92	1704.79	369.59(28)	2998.21(260)	101613.76(1)
ss8ts10nv89sqr150/pnz10	5	328.79	1.65	364.45	348.36(152)	578.82(260)	6517.88(1)
ss8ts10nv89sqr150/pnz15	5	179.20	2.22	137.53	190.61(211)	261.89(260)	1425.35(1)
ss8ts10nv89sqr150/pnz20	5	248.70	2.99	76.59	264.57(363)	213.96(299)	398.79(1)
ss8ts10nv89sqr200/pnz5	5	875.72	1.54	2131.54	930.47(63)	3715.96(260)	106390.15(1)
ss8ts10nv89sqr200/pnz10	5	895.22	2.84	457.30	944.58(374)	600.53(260)	6069.63(1)
ss8ts10nv89sqr200/pnz15	5	387.21	3.82	194.23	411.69(314)	305.69(260)	2172.12(1)
ss8ts10nv89sqr200/pnz20	5	235.49	4.59	116.41	253.32(239)	272.07(260)	2497.59(1)

3.5 Conclusions and future research

In this chapter, we presented an approximation algorithm for solving packing problems that exploits global sparsity structure of the problem and presented a worst case performance analysis of this algorithm. The key takeaways from our computational experiments are:

1. Our algorithm always obtains a better solution than `CPLEX`. Our dual bounds

Table 18: Computational time II (tree instances)

Instance Features	NIns	T^{A}_{AGS}	T^{G}_{AGS}	T^{S}_{AGS}	$T^{G}_{G}(NIt^{G}_{G})$	$T^{G}_{GS}(NIt^{G}_{GS})$	$T^{S}_{GS}(NIt^{S}_{GS})$
ss10ts6nv71sqr100/pnz5	5	168.95	0.42	570.86	179.66(40)	1086.15(260)	22926.51(1)
ss10ts6nv71sqr100/pnz10	5	30.95	0.58	112.29	33.2(35)	198.88(260)	1803.05(1)
ss10ts6nv71sqr100/pnz15	5	51.84	0.92	49.81	55.46(164)	94.83(260)	511.99(1)
ss10ts6nv71sqr100/pnz20	5	54.17	0.98	24.50	57.7(248)	48.62(260)	178.7(1)
ss10ts6nv71sqr150/pnz5	5	506.08	0.72	741.21	534.33(95)	1454.58(260)	33874.55(1)
ss10ts6nv71sqr150/pnz10	5	200.75	1.27	157.81	212.6(202)	245.35(260)	1890.7(1)
ss10ts6nv71sqr150/pnz15	5	130.38	1.75	60.69	138.72(347)	122.55(260)	421.44(1)
ss10ts6nv71sqr150/pnz20	5	155.62	2.06	31.47	165.63(512)	144.77(453)	150.39(2)
ss10ts6nv71sqr200/pnz5	5	1175.88	1.30	1034.42	1239.53(203)	1446.43(260)	34802.64(1)
ss10ts6nv71sqr200/pnz10	5	604.26	1.98	189.08	637.35(574)	296.48(260)	2020.94(1)
ss10ts6nv71sqr200/pnz15	5	276.24	3.03	89.81	294.12(436)	153.59(260)	898.4(1)
ss10ts6nv71sqr200/pnz20	5	163.17	3.69	54.51	175.8(420)	109.44(260)	826.98(1)
ss10ts8nv91sqr100/pnz5	5	204.22	0.52	1249.94	219.91(23)	2715.56(260)	77953.02(1)
ss10ts8nv91sqr100/pnz10	5	51.37	0.75	259.03	55.23(32)	475.27(260)	7561.03(1)
ss10ts8nv91sqr100/pnz15	5	69.41	1.09	104.07	74.48(111)	174.8(260)	1298.2(1)
ss10ts8nv91sqr100/pnz20	5	81.82	1.27	54.10	87.13(207)	109.05(260)	462.51(1)
ss10ts8nv91sqr150/pnz5	5	554.72	0.94	1726.38	590.27(50)	3024.41(260)	104143.85(1)
ss10ts8nv91sqr150/pnz10	5	329.36	1.66	365.60	348.21(157)	529.49(260)	5225.2(1)
ss10ts8nv91sqr150/pnz15	5	169.07	2.11	129.22	180.12(202)	232.93(260)	1173.92(1)
ss10ts8nv91sqr150/pnz20	5	255.44	2.89	75.99	271.98(421)	195.14(260)	264.5(1)
ss10ts8nv91sqr200/pnz5	5	1143.24	1.65	2165.81	1211.79(78)	3682(260)	109987.34(1)
ss10ts8nv91sqr200/pnz10	5	822.01	2.58	417.20	866.83(380)	651.69(260)	7182.31(1)
ss10ts8nv91sqr200/pnz15	5	358.10	3.60	186.68	380.52(289)	304.76(260)	2407.01(1)
ss10ts8nv91sqr200/pnz20	5	271.25	5.17	132.12	291.71(309)	246.04(260)	2501.62(1)
ss10ts10nv111sqr100/pnz5	5	207.69	0.64	2397.74	224.4(11)	5328.36(260)	177250.01(1)
ss10ts10nv111sqr100/pnz10	5	87.04	1.04	527.80	94.58(22)	870.71(260)	14505.78(1)
ss10ts10nv111sqr100/pnz15	5	97.60	1.33	205.81	105.02(80)	331.27(260)	2732.87(1)
ss10ts10nv111sqr100/pnz20	5	113.88	1.57	104.59	121.32(142)	220.43(260)	1477.04(1)
ss10ts10nv111sqr150/pnz5	5	466.18	1.16	3167.85	503.14(21)	5722.86(260)	221493.36(1)
ss10ts10nv111sqr150/pnz10	5	391.15	1.89	636.85	414.28(96)	1103.16(260)	13522.07(1)
ss10ts10nv111sqr150/pnz15	5	231.99	2.59	245.07	247.17(146)	545.41(260)	3776.71(1)
ss10ts10nv111sqr150/pnz20	5	349.08	3.60	145.30	371.03(306)	350.51(260)	645.76(1)
ss10ts10nv111sqr200/pnz5	5	1192.57	1.83	3948.82	1263.85(51)	7122.67(260)	237569.33(1)
ss10ts10nv111sqr200/pnz10	5	1088.86	3.11	779.95	1148.93(232)	1146.4(260)	14088.51(1)
ss10ts10nv111sqr200/pnz15	5	494.80	4.46	354.65	525.93(239)	570.84(260)	5792.69(1)
ss10ts10nv111sqr200/pnz20	5	425.13	6.20	247.42	454.86(205)	510.55(260)	6423.97(1)

are also stronger than those obtained by CPLEX in most cases.

2. Random graph instances: In comparison to GRASP and its variants, the approximation algorithm performs better when the fractional mixed chromatic number is low. In other cases, it provides a good starting solution that can be improved by GRASP.

3. Three-layer tree instances: The approximation algorithm (just on its own) performs particularly well for three-layer tree instances. In particular, GRASP

(even with local search) appears to be significantly less competitive in this case.

The good performance of our algorithm does not imply that powerful and flexible heuristics such as GRASP should be abandoned. Rather the approximation algorithm proposed in this chapter can be viewed as am important tool to exploit known global sparsity pattern for especially large packing IPs so as to produce good solutions efficiently.

We mention here some important directions of future research that need to be explored. Observe that our algorithm is very easily *parallelizable.* In particular, we may solve each of the sub-IPs separately. Even the dual bounding Algorithm 5 will work within this parallel framework. Such a approach can be used to significantly improve the running time of our algorithm. We remark here that parts of GRASP are also parallelizable (for example, run different iterations of basic GRASP in parallel in order to collect a pool of good solutions.)

In order to use advantageously our algorithm for general sparse packing instances for which we do not know the global sparsity structure in advance, an important direction of research is to discover ways to permute the rows and columns of the problems to discover global sparsity structure. Adapting algorithmic ideas/heuristics for problems such as biclustering [23] may be one direction to explore.

Appendix A

PROOF OF THEOREMS (CHAPTER 1)

A.1 Concentration Inequalities

We state Bernstein's inequality in a slightly weaker but more convenient form.

Theorem 84 (Bernstein's Inequality [[27], Appendix A.2]). *Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$ be independent random variables such that $|\boldsymbol{X}_i - \mathbb{E}[\boldsymbol{X}_i]| \leq M$ for all $i \in [n]$. Let $\boldsymbol{X} = \sum_{i=1}^{n} \boldsymbol{X}_i$ and define $\sigma^2 = Var(\boldsymbol{X})$. Then for all $t > 0$ we have*

$$\Pr(|\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}]| > t) \leq \exp\left(-\min\left\{\frac{t^2}{4\sigma^2}, \frac{3t}{4M}\right\}\right).$$

A.2 Empirically Generating Lower Bound on $d(P, P^k)$

We estimate a lower bound on $d(P, P^k)$ using the following procedure. The input to the procedure is the set of points $\{p^1, ...p^t\} \in [0,1]^n$ which are vertices of P. For every $I \in \binom{[n]}{k}$, we use PORTA to obtain an inequality description of $P + \mathbb{R}^{\bar{I}}$. Putting all these inequalities together we obtain an inequality description of P^k. Unfortunately due to the large number of inequalities, we are unable to find the vertices of P^k using PORTA. Therefore, we obtain a lower bound on $d(P, P^k)$ via a *shooting experiment.*

First observe that given $u \in \mathbb{R}^n \setminus \{0\}$ we obtain a lower bound on $d(P, P^k)$ as:

$$\frac{1}{\|u\|}\left(\max\{u^x : x \in P^k\} - \max\{u^x : x \in P\}\right).$$

Moreover it can be verified that there exists a direction which achieves the correct value of $d(P, P^k)$. We generated 20,000 random directions u by picking them uniformly in the set $[-1,1]^n$. Also we found that for instances where $p^j \in \{x \in \{0,1\}^n : \sum_{i=1}^{n} x_i = \frac{n}{2}\}$, the directions $(\frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}})$ and $-(\frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}})$ yield good lower bounds. The Figure in Section 1.3(c) plots the best lower bound among the 20,002 lower bounds found as above.

A.3 Anticoncentration of Linear Combination of Bernoulli's

It is convenient to restate Lemma 8 in terms of Rademacher random variables (i.e. that takes values -1/1 with equal probability).

Lemma 85 (Lemma 8, restated)**.** *Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$ be independent Rademacher random variables. Then for every $a \in [-1,1]^n$,*

$$\Pr\left(a\boldsymbol{X} \geq \frac{\alpha}{\sqrt{n}}\left(1-\frac{1}{n^2}\right)\|a\|_1 - \frac{1}{n^2}\right) \geq \left(e^{-50\alpha^2} - e^{-100\alpha^2}\right)^{60\log n}, \qquad \alpha \in \left[0, \frac{\sqrt{n}}{8}\right].$$

We start with the case where the vector a has all of its coordinates being similar.

Lemma 86. *Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$ be independent Rademacher random variables. For every $\epsilon \geq 1/20$ and $a \in [1-\epsilon, 1]^n$,*

$$\Pr\left(a\boldsymbol{X} \geq \frac{\alpha}{\sqrt{n}}\|a\|_1\right) \geq e^{-50\alpha^2} - e^{-\frac{\alpha^2}{4\epsilon^2}}, \qquad \alpha \in \left[0, \frac{\sqrt{n}}{8}\right].$$

Proof. Since $a\boldsymbol{X} = \sum_i \boldsymbol{X}_i - \sum_i (1-a_i)\boldsymbol{X}_i$, having $\sum_i \boldsymbol{X}_i \geq 2t$ and $\sum_i (1-a_i)\boldsymbol{X}_i \leq t$ implies that $a\boldsymbol{X} \geq t$. Therefore,

$$\begin{aligned}
\Pr(a\boldsymbol{X} \geq t) &\geq \Pr\left(\left(\sum_i \boldsymbol{X}_i \geq 2t\right) \vee \left(\sum_i (1-a_i)\boldsymbol{X}_i \leq t\right)\right) \\
&\geq \Pr\left(\sum_i \boldsymbol{X}_i \geq 2t\right) - \Pr\left(\sum_i (1-a_i)\boldsymbol{X}_i \leq t\right), \qquad (20)
\end{aligned}$$

where the second inequality comes from union bound. For $t \in [0, n/8]$, the first term in the right-hand side can be lower bounded by $e^{-\frac{50t^2}{n}}$ (see for instance Section 7.3 of [32]). The second term in the right-hand side can be bounded using Bernstein's inequality: given that $\mathrm{Var}(\sum_i (1-a_i)\boldsymbol{X}_i) = \sum_i (1-a_i)^2 \leq n\epsilon^2$, we get that for all $t \in [0, n/8]$

$$\Pr\left(\sum_i (1-a_i)\boldsymbol{X}_i \leq t\right) \leq \exp\left(-\min\left\{\frac{t^2}{4n\epsilon^2}, \frac{3t}{4\epsilon}\right\}\right) = e^{-\frac{t^2}{4n\epsilon^2}}.$$

The lemma then follows by plugging these bounds on (20) and using $t = \alpha\sqrt{n} \geq \frac{\alpha}{\sqrt{n}}\|a\|_1$. $\square$

Proof of Lemma 85. Without loss of generality assume $a > 0$, since flipping the sign of negative coordinates of a does not change the distribution of $a\boldsymbol{Z}$ neither the term $\frac{\alpha}{\sqrt{n}}\left(1 - \frac{2}{n^2}\right)\|a\|_1$. Also assume without loss of generality that $\|a\|_\infty = 1$. The idea of the proof is to bucket the coordinates such that in each bucket the values of a is within a factor of $(1 \pm \epsilon)$ of each other, and then apply Lemma 86 in each bucket.

The first step is to trim the coefficients of a that are very small. Define the trimmed version b of a by setting $b_i = a_i$ for all i where $a_i \geq 1/n^3$ and $b_i = 0$ for all other i. We first show that

$$\Pr\left(b\boldsymbol{Z} \geq \frac{\alpha}{\sqrt{n}}\|b\|_1\right) \geq \left(e^{-50\alpha^2} - e^{-100\alpha^2}\right)^{60\log n}, \tag{21}$$

and then we argue that the error introduced by considering b instead of a is small.

For $j \in \{0, 1, \ldots, \frac{3\log n}{\epsilon}\}$, define the jth bucket as $I_j = \{i : b_i \in ((1-\epsilon)^{j+1}, (1-\epsilon)^j]\}$. Since $(1-\epsilon)^{\frac{3\log n}{\epsilon}} \leq e^{-3\log n} = 1/n^3$, we have that every index i with $b_i > 0$ lies within some bucket.

Now fix some bucket j. Let $\epsilon = 1/20$ and $\gamma = \frac{\alpha}{\sqrt{n}}$. Let E_j be the event that $\sum_{i\in I_j} b_i\boldsymbol{Z}_i \geq \gamma\sum_{i\in I_j} b_i$. Employing Lemma 86 over the vector $(1-\epsilon)^j b|_{I_j}$, gives

$$\Pr\left(\sum_{i\in I_j} b_i\boldsymbol{Z}_i \geq \gamma\sum_{i\in I_j} b_i\right) \geq e^{-50\gamma^2|I_j|} - e^{-\frac{\gamma^2|I_j|}{4\epsilon^2}} \geq e^{-50\gamma^2 n} - e^{-\frac{\gamma^2 n}{4\epsilon^2}}, \qquad \gamma \in \left[0, \frac{1}{8}\right].$$

But now notice that if in a scenario we have E_j holding for all j, then in this scenario we have $b\boldsymbol{Z} \geq \gamma\|b\|_1$. Using the fact that the E_j's are independent (due to the independence of the coordinates of $\boldsymbol{Z}$), we have

$$\Pr(b\boldsymbol{Z} \geq \gamma\|b\|_1) \geq \Pr\left(\bigvee_j E_j\right) \geq \left(e^{-50\gamma^2 n} - e^{-\frac{\gamma^2 n}{4\epsilon^2}}\right)^{\frac{3\log n}{\epsilon}}, \qquad \gamma \in \left[0, \frac{1}{8}\right].$$

Now we claim that whenever $bX \geq \gamma\|b\|_1$, then we have $a\boldsymbol{Z} \geq \frac{\alpha}{\sqrt{n}}\left(1 - \frac{2}{n^2}\right)\|a\|_1$. First notice that $\|b\|_1 \geq \|a\|_1 - 1/n^2 \geq \|a\|_1(1 - 1/n^2)$, since $\|a\|_1 \geq \|a\|_\infty = 1$.

Moreover, with probability 1 we have $a\boldsymbol{Z} \geq b\boldsymbol{Z} - 1/n^2$. Therefore, whenever $b\boldsymbol{Z} \geq \gamma\|b\|_1$:

$$a\boldsymbol{Z} \geq b\boldsymbol{Z} - \frac{1}{n^2} \geq \gamma\|b\|_1 - \frac{1}{n^2} \geq \gamma\left(1 - \frac{1}{n^2}\right)\|a\|_1 - \frac{1}{n^2} = \frac{\alpha}{\sqrt{n}}\left(1 - \frac{1}{n^2}\right)\|a\|_1 - \frac{1}{n^2}.$$

This concludes the proof of the lemma. □

A.4 Hard Packing Integer Programs

A.4.1 Proof of Lemma 13

Fix $i \in [n]$. We have $\mathbb{E}[\sum_j \boldsymbol{A}_i^j] = \frac{mM}{2}$ and $\mathrm{Var}(\sum_j \boldsymbol{A}_i^j) \leq \frac{mM^2}{4}$. Employing Bernstein's inequality we get

$$\Pr\left(\sum_j \boldsymbol{A}_i^j < \frac{mM}{2} - \sqrt{m\log 8n}M\right) \leq \exp\left(-\min\left\{\log 8n, \frac{3\sqrt{m\log 8n}}{4}\right\}\right) \leq \frac{1}{8n},$$

where the last inequality uses the assumption that $m \geq 8\log 8n$. Similarly, we get that

$$\Pr\left(\sum_{i,j} \boldsymbol{A}_i^j > \frac{nmM}{2} + \sqrt{nm\log 8n}M\right) \leq \exp\left(-\min\left\{\log 8n, \frac{3\sqrt{nm\log 8n}}{4}\right\}\right) \leq \frac{1}{8n}.$$

Taking a union bound over the first displayed inequality over all $i \in [n]$ and also over the last inequality, with probability at least $1-1/4$ the valid cut $\sum_i(\frac{2}{mM}\sum_j \boldsymbol{A}_i^j)x_i \leq \frac{1}{mM}\sum_{i,j}\boldsymbol{A}_i^j$ (obtained by aggregating all inequalities in the formulation) has all coefficients on the left-hand side being at least $(1 - \frac{2\sqrt{\log 8n}}{\sqrt{m}})$ and the right-hand side at most $\frac{n}{2} + \frac{\sqrt{n\log 8}}{\sqrt{m}}$. This concludes the proof.

A.4.2 Proof of Lemma 14

Fix $j \in [m]$. We have $\mathbb{E}[\sum_{i=1}^n \boldsymbol{A}_i^j] = \frac{nM}{2}$ and $\mathrm{Var}(\sum_{i=1}^n \boldsymbol{A}_i^j) \leq nM^2/4$ and hence by Bernstein's inequality we get

$$\Pr\left(\sum_{i=1}^n \boldsymbol{A}_i^j > \frac{nM}{2} - M\sqrt{n\log 8m}\right) \leq \exp\left(-\min\left\{\log 8m, \frac{3\sqrt{n\log 8m}}{4}\right\}\right) \leq \frac{1}{8m},$$

where the last inequality uses the assumption that $m \leq n$. The lemma then follows by taking a union bound over all $j \in [m]$.

Appendix B

ANALYSIS OF UPPER BOUND ON $Z^{\mathbf{CUT}}$ (CHAPTER 2)

Assume we have the general formulation

$$\begin{aligned} \max \quad & c^T x \\ s.t. \quad & Ax \leq b \\ & x \in \mathbb{B}^n, \end{aligned}$$

where $A \in \mathbb{R}^{m \times n}$. Recall that we are interested in three type of problems: packing, covering and packing with arbitrary constraint matrix. All these categories will be written in the form of the formulation above with different restrictions on A and c. Let $N_i = \{j \in [n] \,|\, A_{ij} \neq 0\}$ be the index set of non-zero entries of i^{th} row of A. Let $\mathcal{N} = \{N_1, N_2, \ldots, N_t\}$, denote $P^{\mathcal{N}} = \bigcap_{i=1}^{t} P^{(N_i)}$ and $z^{cut} = \max_{x \,\in\, P^{\mathcal{N}}} c^T x$.

Our basic strategy is the following: we keep adding cuts on the support of some N_i and checking whether the LP solution will improve. We stop adding cuts when the objective function value does not change, thus obtaining an upper bound on z^{cut}. The formal algorithm is shown as Algorithm 6.

Once we stop adding cut on some N_i, we check whether there is a valid cut on N_{i+1}. The index *count* is the number of groups of supports that adding cuts will not improve the optimal objective function value. Also, as long as adding a cut produces improvement on the objective value, *count* will be reset as 0. The algorithm terminates when one of the following happens:

1. An integral feasible solution is found.

2. The parameter *count* equals to the number of supports t.

Algorithm 6 Estimating z^{cut}

input: $P = \{x|Ax \leq b\}$,$\mathcal{N} = \{N_1, N_2, \ldots, N_t\}$, $z^{old} = -\infty$, $z^{new} = -\infty$, $\epsilon = 10^{-6}$
$i \leftarrow 1$, $count \leftarrow 0$
loop
  Solve $x^* = \text{argmax}_{x \in P} c^T x$,$z^{new} = c^T x^*$
  if x^* is integral **then**
    $z^{cut} = z^{new}$
    Exit Loop
  else if $z^{new} - z^{old} > \epsilon$ **then**
    $z^{old} \leftarrow z^{new}$
    Generate a valid cut $\alpha x \leq \beta$ on the support of N_i based on Algorithm 2
    $P \leftarrow P \bigcap \{x|\alpha x \leq \beta\}$
    $count \leftarrow 0$
  else
    $z^{old} \leftarrow z^{new}$
    if count = t **then**
      $z^{cut} = z^{new}$
      Exit Loop
    else
      $i \leftarrow i + 1(mod\ t)$
      $count \leftarrow count + 1$

In the algorithm, we call a routine to generate the cut on N_i that is formally shown as Algorithm 7. Assume that $\alpha^T x \leq \beta$ is a valid cut on N_i for some i, then $\alpha^T \hat{x} \leq \beta$ holds for all $\hat{x} \in P^I$. However, as the formulation of P^I is implicit, we apply the technique of row generations. Let X be a subset of all integral points in P^I. At the beginning, $X = \emptyset$. And we generate a valid cut (α^*, β^*) for X. Then we solve the following IP

$$\begin{aligned} \max \quad & \alpha^* x - \beta^* \\ s.t. \quad & x \in P^I. \end{aligned}$$

If the optimal value is less or equal to 0 then it means the cut is valid for P^I, otherwise let $X = X \bigcup \{x^*\}$, where x^* is the optimal solution. By re-applying this process, we will either obtain a valid cut or a certificate that no valid cut exists.

Algorithm 7 Cut generation on N_i

Input: $P = \{x|Ax \leq b\}$, $P^I = conv\_hull\{x|x \in P, x \in Z^n\}$, x^*

$X \leftarrow \emptyset$, $\epsilon \leftarrow 10^{-6}$

loop

  Solve $(\alpha^*, \beta^*) = \text{argmax}_{x^T\alpha \leq \beta, \forall x \in X, \|\alpha\|_1 = 1, \text{support of } \alpha = N_i}\ x^{*T}\alpha - \beta$

  if $x^{*T}\alpha - \beta > \epsilon$ **then**

    Solve $x^0 = \text{argmax}_{x \in P^I}\ \alpha^* x - \beta^*$

    if $\alpha^* x^0 - \beta^* > \epsilon$ **then**

      $X \leftarrow X \bigcup \{x^0\}$

    else

      Return (α^*, β^*)

      Exit Loop

    Return $(\alpha, \beta) = (\vec{0}, 0)$

    Exit Loop

Appendix C

DETAILS OF GRASP (CHAPTER 3)

Algorithm 8 is the basic GRASP algorithm from section 3.2.2 of [15]. In this algorithm, the function *RandomSelect*(*alphaSet*, *proba*) picks a random α from the set *alphaSet* according to its assigned probability (stored in set of probabilities called as *proba*). We discuss the choice of these two sets in Section 3.4.4. Similarly, the function *RandomSelect*(RCL) uniformly picks one element from the set of RCL, which would be an index of the variables. The stopping criteria is either number of iterations or a time limit in our experiments, which we discuss in Section 3.4.4.

Algorithm 8 GRASP $(x^*, z) = GRASP(x^0, A, c, alphaSet, proba)$

---

input: $x^0 \in \mathbb{B}^n$; $A \in \mathbb{B}^{m\times n}$; $c \in \mathbb{R}^n$; $alphaSet \in \mathbb{R}^{13}$; $proba \in \mathbb{R}^{13}$.
Let $N = \emptyset$; $z = 0$;$x^* = \vec{0} \in \mathbb{R}^n$;$x^c = \vec{0} \in \mathbb{R}^n$;$\alpha = 0$; $Eval = \vec{0} \in \mathbb{R}^n$; $Limit = 0$; $RCL = \emptyset$.
$\alpha = RandomSelect(alphaSet, proba)$
$Eval_j = c_j / \sum_{i=1}^{m} A_{ij}, \forall j \in [n]$
while (until stopping criteria) **do**
  $x^c = x^0$; $N = [n] \backslash \{j | x^0_j = 1\}$
  while $(N \neq \emptyset)$ **do**
    $Limit = \min_{j\in N}(Eval_j) + \alpha * (\max_{j\in N}(Eval_j) - \min_{j\in N}(Eval_j))$
    $RCL = \{j | j \in N, Eval_j \geq Limit\}$
    $j^* = RandomSelect(RCL)$;$x_{j^*} = 1$
    $N = N\backslash\{j^*\}$; $N = N\backslash\{j | j \in N, \exists i \in [m], A_{ij^*} + A_{ij} \geq 1\}$
  end while
  if $(c^T x^c \geq z)$ **then**
    $x^* = x^c$;$z = c^T x^*$;
  end if
end while
return (x^*, z)

Algorithm 9 shows the sketch of local search procedure and is introduced in section 3.4 of [15]. Specifically, given a solution x^* an $i - j$ *exchange* with x^* is searching for a strictly better solution $\hat{x}$ such that i 1's from x^* turn to 0's in $\hat{x}$ and j 0's from x^* turn to 1's in $\hat{x}$. Every time we find a new solution, the algorithm repeats the 4 searching functions. Since the number of variables of our test instances is large,

Algorithm 9 Local search $(x^*) = LS(x^0, A, c)$

Let $x^* = x^0$.
repeat
  $x^* = searchNeighbourhood(x^*, 0 - 1exchanges)$
  $x^* = searchNeighbourhood(x^*, 1 - 1exchanges)$
  $x^* = searchNeighbourhood(x^*, 2 - 1exchanges)$
  $x^* = searchNeighbourhood(x^*, 1 - 2exchanges)$
until x^* not improved
return x^*

it requires a long time for a single local search. Therefore we do not implement the version (section 3.3 of [15]), which implements local search on every heuristic solution found after the random greedy step. Instead, we implement local search on a few best solutions as discussed in Section 3.4.4.

REFERENCES

[1] Acer, S., Kayaaslan, E., and Aykanat, C., "A recursive bipartitioning algorithm for permuting sparse square matrices into block diagonal form with overlap," *SIAM Journal on Scientific Computing*, vol. 35, no. 1, 2013.

[2] Achterberg, T. (Member of development Team of GUROBI) Personal communication.

[3] Ahmed, S. and Sahinidis, N. V., "An approximation scheme for stochastic integer programs arising in capacity expansion," *Operations Research*, vol. 51, no. 3, pp. 461–471, 2003.

[4] Andersen, K. and Weismantel, R., "Zero-coefficient cuts," in *IPCO*, 2010.

[5] Bansal, N., Korula, N., Nagarajan, V., and Srinivasan, A., "On k-column sparse packing programs," in *Integer Programming and Combinatorial Optimization, 14th International Conference, IPCO 2010, Lausanne, Switzerland, June 9-11, 2010. Proceedings* (Eisenbrand, F. and Shepherd, F. B., eds.), vol. 6080 of *Lecture Notes in Computer Science*, pp. 369–382, Springer, 2010.

[6] Bergner, M., Caprara, A., Furini, F., Lübbecke, M. E., Malaguti, E., and Traversi, E., "Partial convexification of general mips by dantzig-wolfe reformulation," in *Integer Programming and Combinatoral Optimization - 15th International Conference, IPCO 2011, New York, NY, USA, June 15-17, 2011. Proceedings* (Günlük, O. and Woeginger, G. J., eds.), vol. 6655 of *Lecture Notes in Computer Science*, pp. 39–51, Springer, 2011.

[7] Bixby, R. E., Fenelon, M., Gu, Z., Rothberg, E., and R. Wunderling, R., "Mixed-integer programming: a progress report," ch. 18, pp. 309–326, Philadelphia, PA: SIAM, 2004.

[8] Borndörfer, R., Ferreira, C. E., and Martin, A., "Decomposing matrices into blocks," *SIAM Journal on Optimization*, vol. 9, no. 1, pp. 236–269, 1998.

[9] Boyd, S. and Vandenberghe, L., *Convex Optimization.* Cambridge University Press, 2004.

[10] Brooks, R. L., "On colouring the nodes of a network," *Proceedings - Cambridge Philosophical Society. Mathematical and physical sciences*, vol. 37, pp. 194–197, 1941.

[11] Carøe, C. C. and Schultz, R., "Dual decomposition in stochastic integer programming," *Oper. Res. Lett.*, vol. 24, no. 1-2, pp. 37–45, 1999.

[12] Colbourn, C. J. and Dinitz, J. H., *Handbook of Combinatorial Designs, Second Edition*. Boca Raton, FL: CRC press, Taylor and Francis Group, 2006.

[13] Crowder, H., Johnson, E. L., and Padberg, M. W., "Solving large scale zero-one linear programming problem," *Operations Research*, vol. 31, pp. 803–834, 1983.

[14] David, H. and Nagaraja, H., *Order Statistics*. Wiley, 2003.

[15] Delorme, X., Gandibleux, X., and Rodriguez, J., "GRASP for set packing problems," *European Journal of Operational Research*, vol. 153, no. 3, pp. 564–580, 2004.

[16] Dey, S. S., Molinaro, M., and Wang, Q., "Approximating polyhedra with sparse inequalities," *Mathematical Programming*, pp. 1–24, 2015.

[17] Dey, S. S., Lodi, A., Tramontani, A., and Wolsey, L. A., "On the practical strength of two-row tableau cuts," *INFORMS Journal on Computing*, vol. 26, no. 2, pp. 222–237, 2014.

[18] Dey, S. S., Molinaro, M., and Wang, Q., "Analysis of sparse cutting-planes for sparse milps with applications to stochastic milps." arXiv:1601.00198, 2016.

[19] Escoffier, B., Gourvès, L., Monnot, J., and Spanjaard, O., "Two-stage stochastic matching and spanning tree problems: Polynomial instances and approximation," *European Journal of Operational Research*, vol. 205, no. 1, pp. 19–30, 2010.

[20] Fulkerson, D. R. and Gross., O., "Incidence matrices and interval graphs," *Pacific Journal of Mathematics*, vol. 15, pp. 835–855, 1965.

[21] Gu, Z. (Chief Technical Officer and Co-founder of GUROBI) Personal communication.

[22] Guan, Y., Ahmed, S., and Nemhauser, G. L., "Cutting planes for multistage stochastic integer programs," *Operations Research*, vol. 57, no. 2, pp. 287–298, 2009.

[23] Hartigan, J. A., "Direct clustering of a data matrix," *Journal of the American Statistical Association*, vol. 67, no. 337, pp. 123–129, 1972.

[24] Huchette, J., Dey, S. S., and Vielma, J. P., "Beating the SDP bound for the floor layout problem: A simple combinatorial idea." arXiv:1602.07802, 2016.

[25] Kaparis, K. and Letchford, A. N., "Separation algorithms for 0-1 knapsack polytopes," *Mathematical Programming*, vol. 124, no. 1-2, pp. 69–91, 2010.

[26] King, A. D., Lu, L., and Peng, X., "A fractional analogue of brooks' theorem," *SIAM Journal on Discrete Mathematics*, vol. 26, no. 2, pp. 452–471, 2012.

[27] Koltchinskii, V., *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems*. Springer-Verlag, 2011.

[28] Kong, N. and Schaefer, A. J., "A factor 1/2 approximation algorithm for two-stage stochastic matching problems," *European Journal of Operational Research*, vol. 172, no. 3, pp. 740–746, 2006.

[29] Lodi, A., "Mixed integer programming computation," in *50 Years of Integer Programming 1958-2008 - From the Early Years to the State-of-the-Art* (Jünger, M., Liebling, T. M., Naddef, D., Nemhauser, G. L., Pulleyblank, W. R., Reinelt, G., Rinaldi, G., and Wolsey, L. A., eds.), pp. 619–645, Springer, 2010.

[30] Lovász, L., "On the ratio of the optimal integral and fractional covers," *Discrete Mathematics*, vol. 13, pp. 383–390, 1975.

[31] Marchand, H., Martin, A., Weismantel, R., and Wolsey, L. A., "Cutting planes in integer and mixed integer programming," *Discrete Applied Mathematics*, vol. 123, pp. 397–446, 2002.

[32] Matousek, J. and Vondrak, J., *The Probabilistic Method*. 2008. Manuscript.

[33] Molloy, M. and Reed, B., *Graph Colouring and the Probabilistic Method*. Springer-Verlag Berlin Heidelberg, 2002.

[34] Narisetty, A. (Member of development team of SAS IP Solver) Personal communication.

[35] Pritchard, D. and Chakrabarty, D., "Approximability of sparse integer programs," *Algorithmica*, vol. 61, no. 1, pp. 75–93, 2011.

[36] Richard, J.-P. P. and Dey, S. S., "The group-theoretic approach in mixed integer programming," ch. 19, pp. 727–801, New York: Springer-Verlag, 2009.

[37] Walter, M., "Sparsity of lift-and-project cutting planes," in *Operations Research Proceedings 2012* (Helber, S., Breitner, M., Rsch, D., Schn, C., Graf von der Schulenburg, J.-M., Sibbertsen, P., Steinbach, M., Weber, S., and Wolter, A., eds.), Operations Research Proceedings, pp. 9–14, Springer International Publishing, 2012.

[38] Wang, J. and Ralphs, T. K., "Computational experience with hypergraph-based methods for automatic decomposition in discrete optimization," in *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 10th International Conference, CPAIOR 2013, Yorktown Heights, NY, USA, May 18-22, 2013. Proceedings* (Gomes, C. P. and Sellmann, M., eds.), vol. 7874 of *Lecture Notes in Computer Science*, pp. 394–402, Springer, 2013.

[39] Zhang, M. and Küçükyavuz, S., "Finitely convergent decomposition algorithms for two-stage stochastic pure integer programs," *SIAM Journal on Optimization*, vol. 24, no. 4, pp. 1933–1951, 2014.

[40] Ziegler, G. M., *Lectures on Polytopes*. Springer, 1995.